HYBRID NEURAL NETWORK FIRST-PRINCIPLES APPROACH TO PROCESS MODELING

By

SANJAY GUPTA

A DISSERTATION PRESENTED TO THE GRADUATE SCHOOL
OF THE UNIVERSITY OF FLORIDA IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILLOSOPHY

UNIVERSITY OF FLORIDA

1999

This dissertation

is dedicated

to

my parents

ACKNOWLEDGMENTS

I would like to take this opportunity to thank my advisor Dr. Spyros A. Svoronos for his continuing guidance, encouragement and support throughout the course of my Ph.D. He not only guided me to learn new techniques, he was also helpful in showing me the right course in some of the problems in my personal life.

I wish to thank Dr. Hassan El-Shall for his valuable inputs in the chemistry aspect of this project. I would also like to thank my other committee members, Dr. Richard Dickinson, Dr. Oscar Crisalle, and Dr. Ben Koopman, for kindly reviewing my dissertation and serving on my committee.

The friendship and assistance of my colleagues, Pi-Hsin Liu, Robert Bozic, Rajesh Sharma, Dr. Cheng, Dr. Nagui, Rachel Worthen, and Lav Agarwal, will always be valued.

My respect for my parents, brother, and sister for having stood by me and for giving me moral support always kept me motivated to complete this work.

TABLE OF CONTENTS

CHAPTERS

APPENDICES

LIST OF FIGURES

Abstract of Dissertation Presented to the Graduate School
Of the University of Florida in Partial Fulfillment of the
Requirements for Doctor of Philosophy

HYBRID NEURAL NETWORK FIRST-PRINCIPLES APPROACH TO PROCESS MODELING

By

Sanjay Gupta

May 1999

Chairman: Dr. Spyros A. Svoronos
Cochairman: Dr. Hassan El-Shall
Major Department: Chemical Engineering

A hybrid model for a flotation column is presented which combines a first-principles model with artificial neural networks. The first-principles model is derived by making material balances on both phosphate and silica particles in the slurry phase. Neural networks are used to relate the model parameters with operating variables such as particle size, superficial air velocity, frother concentration, collector and extender concentration, and pH. One-level and two-level hybrid modeling structures are compared and it is shown that the two-level structure offers significant advantages over the other. Finally, a sequential run-to run optimization algorithm is developed which combines the hybrid model with an optimization technique. The algorithm guides the changes in the manipulated variables after each experiment to determine the optimal column conditions.

Designed experiments were performed in a lab scale column to generate data for the initial training of the neural networks.

CHAPTER 1
INTRODUCTION

Since the beginning of 1980s, the industrial application of flotation technology has experienced a remarkable growth due to active theoretical and experimental research and development. Flotation columns are slowly being accepted in the mineral processing industry for the advantages they offer over conventional flotation equipment including grade improvement, lower operating cost, and superior control. The ability of flotation columns to produce concentrates of superior grade at similar recovery is derived from the improved selectivity it offers.

Unlike conventional mechanical cells, flotation columns do not use mechanical agitation to suspend particles. Another distinct feature of the flotation column principle is countercurrent contact between feed particles and air bubbles. The lack of moving parts and lower reagent consumption results in a lower operating costs. The lower capital cost for the equipment is attributed to its high capacity leading to the use of less units for the same production rate.

The current flotation practice in Florida phosphate industry involves the use a two stage process with mechanical cells, where the feed is subjected to rougher flotation in which fatty acids and fuel oil are used as collectors to separate the phosphate from most of the sand. The rougher concentrate is then scrubbed by sulfuric acid to remove the fatty acids and oil. The scrubbed material has to be washed with fresh water to achieve a neutral pH. The scrubbed and washed material is then subjected to cleaner flotation in which amine together with kerosene is used as collector to float sand. This stage of flotation is sensitive to impurities in water; thus, fresh water is used in most of the plants as make up water. However, the fatty acid circuit uses recycled water. This process has become less cost effective due to high cost of reagents and increasing concentration of contaminants.

To prepare the phosphate feed, the mined phosphate ore (matrix) is washed and de-slimed at 150 mesh. The material finer than 150 mesh is pumped to clay settling ponds. The rock coarser than 150 mesh is screened to separate pebbles (-3/4 +14 mesh) which are of high phosphate content. Washed rock (-14, +150 mesh) is sized into a fine (usually 35 x 150 mesh) and a coarse flotation feeds (usually 14 x 35 mesh) which are treated in separate circuits. Flotation of phosphates from the fine feed (35 X 150 mesh) presents very few difficulties and recoveries in excess of 90% are achieved using conventional flotation cells. On the other hand, recovery of phosphate values from the coarse feed is much more difficult and flotation by itself usually yields recovery of 60% or less.

The density of the solid, turbulence, stability and height of the froth layer, depth of the water column, viscosity of the froth layer are known to effect the flotation process in general (Boutin and Wheeler, 1967). However, the exact reasons for low recovery of coarse particles in conventional flotation is not very well understood. There are several hypotheses about the flotation behavior of coarse particles. For instance, the floatability of large particles could be due to the additional weight that has to be lifted to the surface under the heavy turbulence conditions, and the difficulty to transfer and maintain these particles in the froth layer. Some efforts towards improving the flotation of coarse particles through stabilization of the froth layer, minimizing the froth height, and addition of an elutriation water stream at the bottom of the column have been undertaken.

The equipment used by the phosphate industry in flotation process are not selective enough to take full advantage of new reagents and operating schemes, to recover phosphate from the coarse feed or to optimize results with existing reagents. The best way to increase the selectivity of phosphate flotation is to improve upon the design of flotation equipment. Particularly the new equipment should improve the recovery of coarse particles, while still providing the high selectivity of fine particles.

It has been found both theoretically and practically that flotation columns have better separation performance than conventional mechanical cells (Finch and Dobby, 1990). The use of flotation columns can not only help overcome some of the problems related to coarse phosphate flotation but it has several other advantages as mentioned above. Spargers or bubble generating systems are the single most important element in the flotation columns. They are generally characterized in terms of their air dispersion ability. Frothers are the chemicals that help in controlling and stabilizing bubble size by reduction of surface tension. Thus both of them play an important role in the overall performance of flotation columns. Their interaction can be a crucial factor in the success of flotation column.

Flotation columns have been used predominantly in the coal beneficiation industry. However, their application in other mineral industries, such as the phosphate, is not very well studied. Unlike other minerals, phosphate flotation deals with a considerably larger size of particles (0.1-1mm) and therefore the operation of phosphate flotation in a column is different from that of other minerals. High recovery and grade and low operating cost depend largely on the optimal selection of operating variables such as the air flow rate, the frother type and concentration, and the elutriation water rate. The search of the optimal conditions can considerably benefit by the availability of a model that can predict the effects of different operating conditions on column behavior. Finch and Dobby (1990) and Lutrell and Yoon (1993) developed a one-phase axial dispersion model in which particle collection is viewed as a first order net attachment rate process. Sastry and Loftus (1988) considered both the slurry and air phases and they used two separate first order rate constants for attachment and detachment of the particles. However, these models cannot predict the effects of certain operating conditions such as particle size, frother concentration, collector and extender concentration, and pH on the flotation performance.

In this work, a mathematical model is developed that for the first time predicts the effects of particle size, frother concentration, collector and extender concentration, and pH on the flotation behavior. This is a hybrid model that combines a first-principles model with artificial neural networks (ANNs). The first-principles model is derived by making a material balance on solid particles in the slurry phase. First order reaction rate constants are assumed for the attachment of the solid particles to the air bubbles. Single output feedforward backpropagation neural networks are used to correlate the model parameters with the operating variables.

Two hybrid modeling approaches are presented. Chapter 2 describes a one-level hybrid model that uses three different neural networks to predict the flotation rate constant for phosphate, the flotation rate constant for gangue, and air holdup. Chapter 3 presents a two-level hybrid model in which neural networks are structured in two levels. Two neural networks are used in the top-level to predict bubble diameter and air holdup. The bubble diameter is used as an input in the neural networks of the bottom-level which predict the flotation rate constants for phosphate and gangue. The inherent advantages and disadvantages of the two hybrid modeling approaches are also discussed in these chapters.

In chapter 4, the hybrid model developed is combined with an on-line optimization algorithm to determine the optimal conditions for column operation. The algorithm guides successive changes of the manipulated variables such as air flow rate, frother concentration, and pH, after each run to achieve optimal column operating conditions. Designed experiments were performed to generate data for the initial training of the neural networks. The trained neural network is then used to guide the direction of the new experiments.

CHAPTER 2
ONE-LEVEL HYBRID MODEL

Flotation is a process commonly employed for the selective separation of phosphate from unwanted mineral. Column flotation is slowly gaining popularity in the mineral processing industry, including the phosphate industry, due to its ability to improve selectivity, lower operating cost, lower capital cost, and superior control. In this work, a hybrid model is developed that combines a physicochemical model with artificial neural networks. This model for the first time incorporates the effect of collector concentration, extender concentration, and pH on the flotation performance. The physicochemical model is based on axial dispersion with first order collection rates. Three basic parameters are required in this model: flotation rate constant for phosphate, flotation rate constant for gangue, and air holdup. Artificial neural networks are used to predict these parameters. The model also takes into account the particle size distribution and predicts grade and recovery for each particle size range. The model is validated against laboratory column data.

2.1 Introduction

Even though the concept of column flotation was developed (Wheeler, 1988) and patented (Boutin and Wheeler, 1967) in the early 1960s, its acceptance for the processing and beneficiation of phosphate ores is relatively recent. The majority of the phosphate plants employ mechanical cells. However, column flotation has simpler operation and provides superior grade/recovery performance. For these reasons column flotation is gaining increasing acceptance for the processing and beneficiation of phosphate ores. Although it has been successfully employed for the selective separation of phosphate from unwanted mineral, a totally predictive model still remains unavailable for industrial use.

Flotation is a process to separate hydrophobic particles from hydrophilic particles. The hydrophobic material has a tendency to attach to the rising bubbles and leaves from the top of the column. The hydrophilic material settles down and leaves from the bottom of the column. In this way, the phosphate containing material (frankolite or apatite) is separated from gangue (mostly silica). The phosphate ore is first pretreated with fatty acid collector and fuel oil extender. Fatty acid and fuel oil adsorb on the phosphate-containing particles rendering them hydrophobic. The flotation process is then used to separate phosphate particles from gangue minerals.

A flotation column consists of three flow regimes: a cleaning or froth zone, a lower collection zone, and pulp-froth interface zone. The froth zone is the region extending upward from the pulp-froth interface to the column interface. The collection zone is the region extending downward from the pup-froth interface to the lowest sparger. A mineral particle is recovered by a gas bubble in the collection zone of the column by particle-bubble collision followed by attachment due to the hydrophobic nature of the mineral surface. Since phosphate particles are considerably larger in size (0.1-1 mm), an elutriation water stream from the bottom is added to maintain a positive upward flow (negative bias) to aid lifting the particles upward.

The particle collection process in a column is considered to follow first order kinetics relative to the solids particle concentration with a rate constant. Finch and Dobby (1990) and Lutrell and Yoon (1993) used a one-phase axial dispersion model in which particle collection is viewed as a first order net attachment rate process. Sastry and Loftus (1988) considered both the slurry and air phases and they used two separate first order rate constants for attachment and detachment of the particles. Luttrell and Yoon (1993) relate the particle net attachment rate constant to some operating variables using a probabilistic approach. However, their approach cannot be used to predict the effect of certain operating conditions such as frother concentration, collector concentration, extender concentration, and pH.

For the model to be predictive, the functional dependence of the net attachment rate constant (k_p or k_g) on the key operating variables needs to be determined. The functional relationship of model parameters on the operating conditions is difficult to determine via physicochemical reasoning. In our approach, we use neural networks to determine these functional relationships. Artificial neural networks are a powerful tool, inspired by how the human brain works, that can learn from examples any unknown functional relationship. Their ability to approximate any smooth nonlinear multivariable function arbitrarily well (Hornik *et al.*, 1989) and their simple construction have led to great interest in using neural networks.

Existing modeling strategies can be divided into white-box, black-box, and gray-box (hybrid) strategies, depending on the amount of prior knowledge that is used for development of the model. White-box modeling strategies are mainly knowledge driven.

Black-box modeling strategies are mainly data driven and the resulting models often do not have reliable extrapolation properties. Black-box strategies have been applied to many chemical processes, especially since convenient black-box modeling tools like neural networks have become available (Bhat and McAvoy, 1990; Psichogios and Ungar, 1992a). Gray-box or hybrid modeling strategies are potentially very efficient if the black-box and white-box components are combined in such a way that the resulting models have good interpolation and extrapolation properties.

There are two types of gray-box modeling approaches in which a neural network is combined with a black-box model: the parallel and the serial approach. In the parallel approach, the neural network is placed parallel with a white-box model. In this case, the neural network is trained on the error between the output of the white-box model and the actual output. Su *et al.* (1992) demonstrated that the parallel approach resulted in better interpolation properties than pure black-box models. Johansen and Foss (1992) also used a parallel structure where the output of the hybrid model was a weighted sum of a first-principles and a neural network model.

In the serial hybrid modeling strategy, the neural network is placed in series with the first-principles model. Various researchers (Psichogios and Ungar, 1992a; Thompson and Kramer, 1994) have shown the potential extrapolation properties of serial hybrid models. Psichogios and Ungar (1992b) used this approach for parameters that are functions of the state variables and manipulated inputs. Liu *et al.* (1995) developed a serial hybrid model for a periodic wastewater treatment process by using ANNs for the bio-kinetic rates of a first-principles model. Cubillo and Lima (1997) also used this approach to develop hybrid model for a rougher flotation circuit.

In this work, we employ a serial approach to integrate an approximate model, derived from first-principles considerations, with neural networks which approximates the unknown kinetics. The first-principles model is inverted to calculate two model parameters for each set of measured recovery and grade. The neural networks are then trained on the errors of calculated model parameters instead of the errors of the output of the first-principles model as is the case with the above referenced works. Also, unlike most other cited work, we employ experimental data instead of simulated data.

2.2 First-Principles Model

The basic equations representing the flotation of solid particles in a flotation column can be written by making a material balance for the solid particles in the slurry phase. This results in the following partial differential equations for the section above and below the feed point, respectively:

$$\frac{\partial C_{p_1}^{j}}{\partial t} = \left(\frac{U_p}{1-\varepsilon_g} - U_{sl}^{j} \right) \frac{\partial C_{p_1}^{j}}{\partial z} + D \frac{\partial^2 C_{p_1}^{j}}{\partial z^2} - k_p(d_p^j) C_{p_1}^{j} \qquad (2.1)$$

$$\frac{\partial C_{p_2}^{j}}{\partial t} = \left(\frac{U_t}{1-\varepsilon_g} + U_{sl}^{j} \right) \frac{\partial C_{p_2}^{j}}{\partial z} + D \frac{\partial^2 C_{p_2}^{j}}{\partial z^2} - k_p(d_p^j) C_{p_2}^{j} \qquad (2.2)$$

where

$C^{j}_{p_1}$ = Phosphate concentration of jth mesh size particles for the section above the feed point

$C^{j}_{p_2}$ = Phosphate concentration of jth mesh size particles for the section below the feed point

U_p = Superficial liquid velocity above the feed point

= Q_p / A_c

U_t = Superficial liquid velocity below the feed point

= $(Q_t - Q_e)/A_c$

D = Dispersion coefficient

Q_p = Product volumetric flow rate

Q_t = Tailings volumetric flow rate

Q_e = Elutriation volumetric flow rate

A_c = Cross-sectional area of the column

U^{j}_{sl} = Slip velocity of jth mesh size particles

ε_g = Air holdup

$k_p(d^{j}_{p})$ = Flotation rate constant for phosphate for jth mesh size particles

The following assumptions are made in deriving the above equations:

1) The concentration of solid particles in the slurry phase is a function of height, z only, and variations of the concentration in radial and angular directions can be neglected.

2) The air holdup is constant throughout the column.

3) All the air bubbles in the system are of a single size.

4) Rate of detachment is either negligible or is a function of conditions in the slurry phase. This assumption allows to treat the net attachment rate with just one floatation rate constant.

The slip velocity is calculated using the expression of Villeneuve *et al.* (1996):

$$U_{sl}^{j} = \frac{g\, d_p^{j^2} (\rho_s - \rho_l)(1 - \phi_s)^{2.7}}{18\mu_l (1 + 0.15 R_{ep}^{j^{0.687}})} \tag{2.3}$$

where the particle Reynolds number is defined as

$$R_{ep}^{j} = \frac{d_p^{j}\, U_{sl}^{j} \rho_s (1 - \phi_s)}{\mu_l} \tag{2.4}$$

where

g = Acceleration due to gravity (m/s^2)

μ_l = Water viscosity (kg/ms)

ρ_l = Water density (kg/m^3)

ρ_s = Solid density (kg/m^3)

ϕ_s = Volume fraction of solids in slurry

d_p^{j} = Particle diameter (m)

Since R_{ep}^{j} is a function of U_{sl}^{j}, an iterative procedure is used to calculate the slip velocity. The procedure starts with an initial guess for U_{sl}^{j} and corresponding value of R_{ep}^{j} is plugged in Equation 2.3 and new value of U_{sl}^{j} is found. This new value is then used in Equation 2.2 and this procedure is continued till convergence is achieved. The axial dispersion coefficient is calculated by a modified expression of Finch and Dobby (1990):

$$D = 0.063\,(1 - \varepsilon_g)\, d_c \left(\frac{J_g}{1.6}\right)^{0.3} \tag{2.5}$$

where

d_c = column diameter (m)

J_g = superficial air velocity (cm/s)

Equations 2.1 and 2.2 can be solved analytically for the concentration profile of the solid particles at steady state. The resulting analytical expressions for the concentration profile are

$$C^j_{p_1} = K^j_1 \exp\{-\frac{1}{2}(a^j - \sqrt{a^{j^2} + 4b^j})z\} + K^j_2 \exp\{-\frac{1}{2}(a^j + \sqrt{a^{j^2} + 4b^j})z\} \qquad (2.6)$$

$$C^j_{p_2} = K^j_3 \exp\{-\frac{1}{2}(d^j - \sqrt{d^{j^2} + 4b^j})z\} + K^j_4 \exp\{-\frac{1}{2}(d^j + \sqrt{d^{j^2} + 4b^j})z\} \qquad (2.7)$$

where

$$a^j = \frac{\left(\frac{U_p}{1-\varepsilon_g} - U^j_{sl}\right)}{D}; \quad b^j = \frac{\left(\frac{U_t}{1-\varepsilon_g} + U^j_{sl}\right)}{D}; \quad \text{and } d^j = \frac{k_p(d^j_p)(1-\varepsilon_g)}{D}$$

K^j_1, K^j_2, K^j_3, and K^j_4 are the constants of integration to be determined by using appropriate boundary conditions.

2.2.1 Boundary Conditions

A material balance at the top layer of the column (z = L) gives the following equation:

$$A_c \Delta z \frac{dC^j_{p_1}}{dt} = A_c\left(\frac{U_p}{1-\varepsilon_g} - U^j_{sl}\right)\left(C^j_{p_2} - C^j_{p_1}\right) + A_c D \frac{C^j_{p_2} - C^j_{p_1}}{\Delta z} - k_p(d^j_p) A_c \Delta z C^j_{p_1}$$

(2.8)

in the limit as Δz → 0, the above equation reduces to the following boundary condition:

$$\left.\frac{dC^j_{p_1}}{dz}\right|_{z=L} = 0 \qquad (2.9)$$

Continuity of the concentration profile at the feed location gives

$$\left.C^j_{p_1}\right|_{z=L_f} = \left.C^j_{p_2}\right|_{z=L_f} \qquad (2.10)$$

A similar material balance at the feed inlet gives for the solid particles in the slurry phase

$$Q_f C^j_f = A_c\left(\frac{U_p}{1-\varepsilon_g} - U^j_{sl}\right)\left.C^j_{p_1}\right|_{z=L_f} - A_c D\left.\frac{dC^j_{p_1}}{dz}\right|_{z=L_f} + A_c\left(\frac{U_t}{1-\varepsilon_g} + U^j_{sl}\right)\left.C^j_{p_2}\right|_{z=L_f} + A_c D\left.\frac{dC^j_{p_2}}{dz}\right|_{z=L_f} \qquad (2.11)$$

where

C^j_f = Phosphate feed concentration of jth mesh size particles

Q_f = Feed volumetric flow rate

L_f = Feed location

At the bottom of the column (z = 0), due to the elutriation flow, the derivative of the concentration profile reduces to the following expression:

$$D\left.\frac{dC^j_{p_2}}{dz}\right|_{z=0} = -\frac{Q_e}{(1-\varepsilon_g)A_c}\left.C^j_{p_2}\right|_{z=0} \qquad (2.12)$$

The four boundary conditions can be solved in conjunction with Equations 2.6 and 2.7 for K_1^j, K_2^j, K_3^j, and K_4^j. The resulting expressions for the constants of integration are given by the following equations:

$$K_4^j = \frac{\left(Q_f C_f^j / A_c D\right)}{m^j(a^j - \alpha^j)p^j \exp\{\alpha^j L_f\} + (d^j - \gamma^j)q^j \exp\{\gamma^j L_f\} + m^j(a^j - \beta^j)\exp\{\beta^j L_f\} + (d^j - \delta^j)\exp\{\delta^j L_f\}} \tag{2.13}$$

$$K_3^j = q^j K_4^j \tag{2.14}$$

$$K_2^j = m^j K_4^j \tag{2.15}$$

$$K_1^j = p^j m^j K_4^j \tag{2.16}$$

where

$$\alpha^j = -\frac{a^j}{2} + \frac{1}{2}\sqrt{a^{j^2} + 4b^j} \tag{2.17}$$

$$\beta^j = -\frac{a^j}{2} - \frac{1}{2}\sqrt{a^{j^2} + 4b^j} \tag{2.18}$$

$$\gamma^j = -\frac{d^j}{2} + \frac{1}{2}\sqrt{d^{j^2} + 4b^j} \tag{2.19}$$

$$\delta^j = -\frac{d^j}{2} - \frac{1}{2}\sqrt{d^{j^2} + 4b^j} \tag{2.20}$$

$$p^j = \frac{\left\{-\dfrac{Q_p}{A_c D} + a^j - \beta^j\right\}\exp(\beta^j L)}{\left\{\dfrac{Q_p}{A_c D} - a^j + \alpha^j\right\}\exp(\alpha^j L)} \tag{2.21}$$

$$q^j = \frac{\left\{-\dfrac{Q_t}{A_c D} + d^j - \delta^j\right\}}{\left\{\dfrac{Q_t}{A_c D} - d^j + \gamma^j\right\}} \tag{2.22}$$

$$m^j = \frac{q^j \exp(\gamma^j L_f) + \exp(\delta^j L_f)}{p^j \exp(\alpha^j L_f) + \exp(\beta^j L_f)} \tag{2.23}$$

The algorithm for solving the first-principles model is given in Appendix A.

2.2.2 Calculation of Recovery and Grade

Recovery (%) is defined as the ratio of the weight of the phosphate in the concentrate stream to the weight of the phosphate in the feed stream. The recovery of the phosphate particles of the jth mesh size can be expressed in terms of the feed and tailings flow rates and concentration as

$$R_p^j = \left(\frac{Q_f C_f^j - \left[Q_t + A_c(1-\varepsilon_g)U_{sl}^j\right] C_{p_2}^j \Big|_{z=0}}{Q_f C_f^j} \right) * 100 \tag{2.24}$$

Grade, a measure of the quality of the product, is defined as the ratio of the weight of the phosphate to the total weight recovered in the concentrate stream. Grade is reported as % Bone Phosphate of Lime (% BPL) which is the equivalent grams of tricalcium phosphate

$Ca_3(PO_4)_2$ in 100g of sample. Grade can be calculated as the ratio of the weight of phosphate to the sum of the weight of the phosphate and gangue in the concentrate stream:

$$G^j = \left(\frac{Q_f C_f^j - \left[Q_t + A_c(1-\varepsilon_g)U_{sl}^j\right] C_{p_2}^j \Big|_{z=0}}{(Q_f C_f^j - [Q_t + A_c(1-\varepsilon_g)U_{sl}^j] C_{p_2}^j \Big|_{z=0}) + (Q_f C_{f_g}^j - [Q_t + A_c(1-\varepsilon_g)U_{sl}^j] C_{g_2}^j \Big|_{z=0})} \right) * 73.3$$

(2.25)

where $C_{g_2}^j$ is the gangue concentration of the j^{th} particle size and $C_{f_g}^j$ is the gangue feed concentration of j^{th} particle size. The multiplication factor is 73.3 instead of 100, because pure Florida phosphate rock measures at about 73.3 %BPL.

2.2.3 Model Parameters

The above model formulation has only two model parameters, namely, the flotation rate constants for phosphate and gangue. The experimental analysis in the industry is usually available in terms of grade and recovery of phosphate. The recovery of gangue can then be readily calculated from grade and recovery of phosphate using the following relationship:

$$R_g^j = \frac{R_p^j G_f^j (73.3 - G^j)}{G^j (73.3 - G_f^j)} \qquad (2.26)$$

where G_f^j is the grade of the feed material.

The recovery of phosphate R_p^j is only a function of the flotation rate constant for phosphate, k_p, and air holdup, ε_g. Similarly, the recovery of gangue R_g^j is only a function of flotation rate constant for gangue, k_g, and air holdup, ε_g. Since air holdup is measured, we can invert the model to determine the value of k_p that results in the measured recovery of phosphate R_p^j and the value of k_g that yields the measured recovery of gangue R_g^j. As shown in Figure 2.1, a one-dimensional search is performed to determine the values of flotation rate constants when supplied with the recovery of phosphate and gangue, respectively. This algorithm allows determination of the flotation rate constants for each run, given the operating conditions and the performance of the column in terms of grade and recovery. The algorithm requires two initial guesses of the flotation rate constants which yield errors in the corresponding R_p^j of opposite sign, and then the program uses the method of false position (Chapra and Canale, 1988) to determine the correct set of flotation rate constants.

Recovery of phosphate increases monotonically with flotation rate constant for phosphate, k_p. This is verified by calculating recovery for different values of flotation rate constant and recovery was plotted against flotation rate constant. From the graph shown in Figure 2.2, it is concluded that there is only value of floatation rate constant for a given recovery. Similarly, from Figure 2.3, it is concluded that recovery of gangue increases monotonically with flotation rate constant for gangue, k_g.

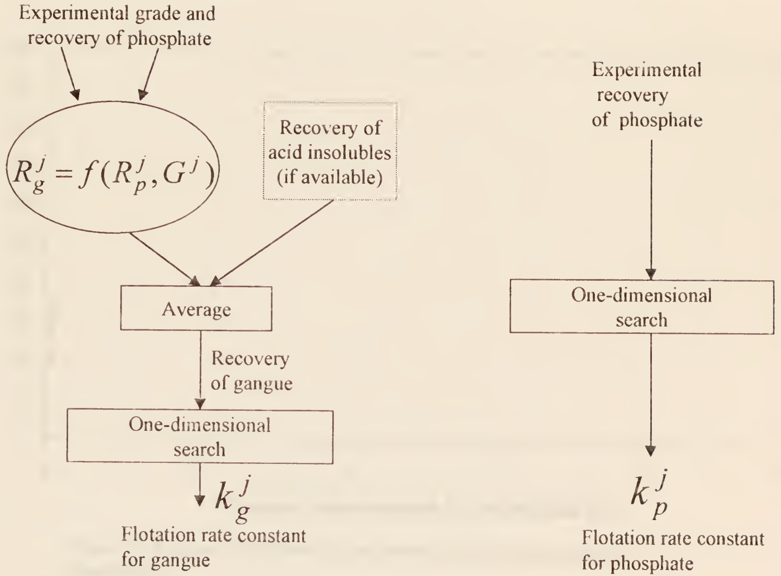
Experimental grade and recovery of phosphate
$R_g^j = f(R_p^j, G^j)$
Recovery of acid insolubles (if available)
Average
Recovery of gangue
One-dimensional search
k_g^j
Flotation rate constant for gangue
Experimental recovery of phosphate
One-dimensional search
k_p^j
Flotation rate constant for phosphate

Figure 2.1: Flotation rate constants for phosphate and gangue are calculated by using a one-dimensional search to invert the first-principles model

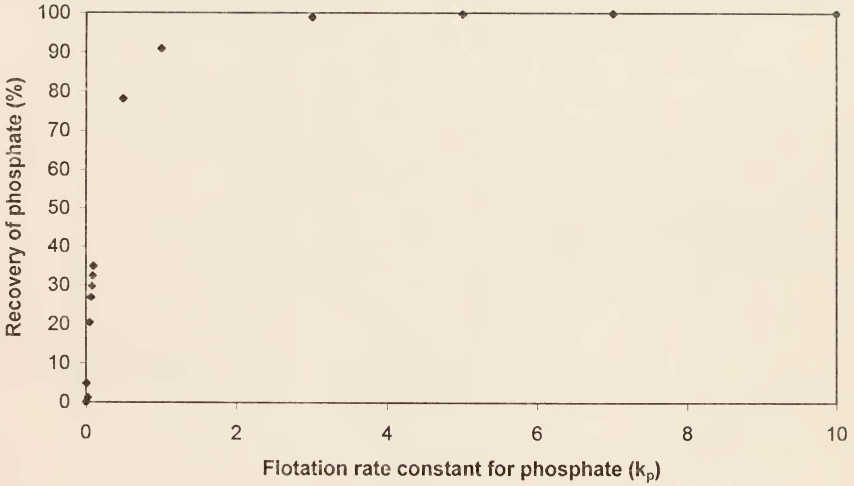

Figure 2.2: Recovery of phosphate (%) as a function of flotation rtae constant for phosphate (k_p)

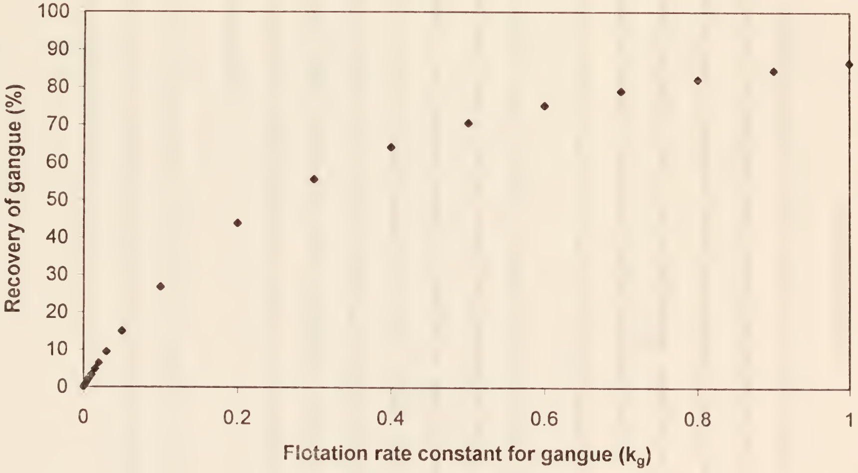

Figure 2.3: Recovery of gangue (%) as a function of flotation rtae constant for gangue (k_g)

2.3 The Hybrid Model

The overall structure of the hybrid model is shown in the Figure 2.4. The hybrid model utilizes backpropagation neural networks (Rumelhart and McClelland, 1986) to predict the values of parameters flotation rate constants, k_p and k_g, and air holdup, ε_g. The factors that affect k_p and k_g are particle diameter, superficial air velocity, frother concentration, collector concentration, extender concentration, and pH. The air holdup, ε_g, is mainly affected by superficial air velocity and frother concentration.

The hybrid model of Figure 2.4 integrates the first-principles model with three artificial neural networks. Neural network, NNI, correlates the flotation rate constant for phosphate, k_p, with phosphate particle size, superficial air velocity, frother concentration, collector concentration, extender concentration, and pH. Similarly, neural network, NNII correlates the flotation rate constant for gangue, k_g, with gangue particle size, superficial air velocity, frother concentration, collector concentration, extender concentration, and pH. Neural network NNIII correlates the air holdup, ε_g, with superficial air velocity and frother concentration.

In this structure, all three neural networks are specific to the type of frother or sparger used. This necessitates generation of new data and retraining of the neural networks each time the frother or the sparger are changed.

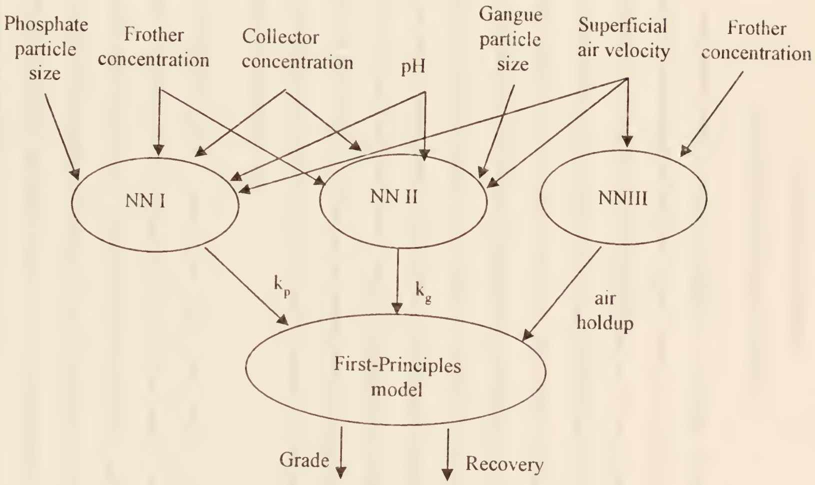

Figure 2.4: Overall structure of the hybrid model

2.4 Materials and Methods

2.4.1 Experimental setup and Procedures

The experimental setup is shown in Figure 2.5. It includes an agitated tank (conditioner) for reagentizing the feed and a screw feeder for controlling the rate of reagentized feed to the flotation column. The agitated tank was 45 cm in diameter and 75 cm high. It was equipped with an impeller of two axial type blades (each 28 cm diameter) The impeller rotation speed was fixed at 465 rpm. The impeller had about 3.8 cm clearance from the bottom of the tank. The feeder with 2.5 cm diameter screw delivered the conditioned phosphate materials to the column. The feed rate was controlled by adjusting the screw rotation speed. Flotation tests were conducted using a 14.5 cm diameter by 1.82 m high plexiglass flotation column. The feed inlet was located at 30 cm from the column top. The discharge flow rate was controlled by a discharge valve and an adjustable speed pump. Three flowmeters were used to monitor the flow rates for air, frother solution, and elutriation water.

Three different feed sizes obtained from Cargill were used in the flotation experiments: coarse feed with narrow distribution (14X35 Tyler mesh), fine feed with wide size distribution (35X150 Tyler mesh), and unsized feed which is a mixture of the above two (14X150 Tyler mesh). For each run, 50 kg of feed sample was added in the pre-treatment tank and water was added to obtain 72% solids concentration by weight. The feed material was then agitated for 10 seconds. 10 % soda ash solution was added to the pulp to reach pH of about 9.4 and agitated for 10 seconds. Subsequently a mixture of

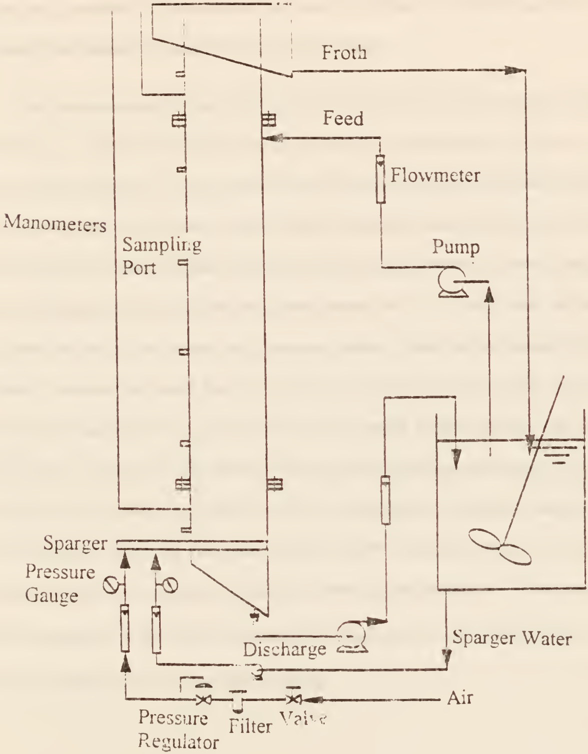

Figure 2.5 A schematic diagram of the experimental setup

fatty acid (obtained from Westvaco) and fuel oil (No. 5 obtained from PCS Phosphate) with a ratio of 1:1 by weight was added to the pulp. The total conditioning time was 3 minutes. The conditioned feed material (without its conditioning water) was loaded in the feeder bin located at the top of the column.

The frother selected for this study was CP-100 (sodium alkyl ether sulfate obtained from Westvaco). Frother-containing water and air were first introduced into the column through the sparger (eductor) at a fixed flowrate and frother concentration, and then the discharge valve and pump were adjusted to get the desired underflow and overflow rates. Air holdup was measured for the two-phase (air/water) system using a differential pressure gauge. After every parameter was set and the two-phase system was in a steady state, the phosphate material was fed to the column using the screw feeder. Water was also added to the screw feeder to maintain the steady flow of the solids to the column at 66 % solids concentration. To achieve steady state, the column was run for a period of three minutes with phosphate feed prior to sampling. Timed samples of tailings and concentrates were taken. The collected samples were weighed and analyzed for %BPL according to the procedure recommended by the Association of Florida Phosphate Chemists (AFPC Analytical Methods, 1980). These measurements were then used to calculate recovery of acid insolubles. These values were then averaged with the values obtained from Equation 2.26 to obtain the R_g^j used to determine the flotation rate constants of gangue.

2.4.2 Experimental Conditions

For the frothers investigated, 35 three-phase experiments were conducted. Seven different levels of frother concentration (5, 6.6, 10, 15, 20, 23.4, and 25 ppm) was studied in designed experiments. Five different levels of collector and extender concentration (0.27, 0.41, 0.54, 0.64, and 1.7 kg/t) were used. pH was varied from 8.2 to 9.9 at five different levels (8.2, 8.5, 9.0. 9.5, and 9.9). Two superficial air velocities (0.46 and 0.7 cm/s) were used for the designed experiments.

The particle size depended on the type of feed used. For coarse feed, the particle size varied from 417 to 991 microns. For fine feed, the particle size varied from 104 to 417 microns whereas for the unsized feed distribution, the size ranged from 104 to 991 microns.

2.4.3 Neural Network Structure and Training

Single output feedforward backpropagation neural networks are used with a single layer of hidden nodes. A unit bias is connected to both the hidden layer and the output layer. Both the hidden layer and the output layer used a logistic activation function (Hertz *et al.*, 1992) and the input and the output values were scaled from 0 to 1.

During the training mode, training examples are presented to the network. A training example consists of scaled input and output values. For NNI and NNII, the output values are the flotation rate constants calculated from one-dimensional searches for phosphate and gangue, respectively. For NNIII, the output value is the experimentally measured air holdup.

The training process is started by initializing all weights randomly to small non-zero values. The random number is generated between -3.4 and +3.4 with standard deviation of 1.0 following the procedure recommended by Masters (1993). The optimal weights were determined using simulated annealing (Kirkpatrick *et al.*, 1983) and a conjugate gradient algorithm (Polak, 1971). There are two approaches towards updating the weights. In one approach, the input-output examples are presented one at a time and after each presentation the weights are updated using rules such as the delta rule (Rumelhart and McClelland, 1986). This method is attractive for its simplicity but is restricted to rather primitive optimization algorithms. In contrast, the batch training approach allows use of powerful methodology for nonlinear optimization. It processes each input-output example individually but updates the weights only after the whole set of input-output examples has been processed. In this case, the gradient is cumulated for all presentations, then the weights are updated, and finally the sum of the squared errors is calculated.

The simulated annealing algorithm is used for eluding local minimum. It perturbs the independent variables (the weights) while keeping track of the best (lowest error) function value for each randomized set of variables. This is repeated several times, each time decreasing the variance of the perturbations with the previous optimum as the mean. The conjugate gradient algorithm is then used to minimize the mean-squared output error. When the minimum is found, simulated annealing is used to attempt to break out of what may be a local minimum. This alteration is continued until networks can not find any lower point. We then hope that the local minimum is indeed the global minimum.

2.5 Results and Discussion

The performances of the three ANNs are shown in Figures 2.6-2.14. Figure 2.6 compares the flotation rate constants for phosphate (k_p) determined from one-dimensional searches with those predicted by NNI. As shown in this figure, NNI captures the dependence of the flotation rate constant on particle size, superficial air velocity, frother concentration,

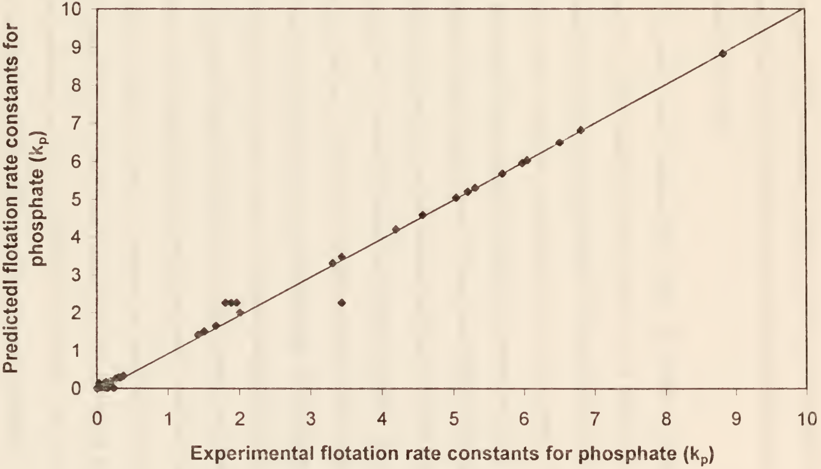

Figure 2.6 Performance of NNI: Model versus experimental flotation rate constant for phosphate (k_p)

collector and extender concentration, and pH. Similarly, Figure 2.7 compares flotation rate constant for gangue (k_g) determined from one-dimensional searches with those predicted by NNII. As shown, NNII successfully predicts the flotation rate constant for gangue. Figure 2.8 presents the air holdup (ε_g) predicted using NNIII against those measured experimentally. A satisfactory match is seen.

The hybrid model integrates NNI, NNII, and NNIII as shown in Figure 2.4. Predictions of the hybrid model are shown in Figure 2.9-2.14. Figures 2.9 and 2.10 compare the experimental recovery (%) and grade (%BPL) with those predicted by the hybrid model, respectively, for the coarse feed size distribution (14X 35 Tyler mesh). As shown in these figures, the hybrid model successfully predicts both recovery and grade. Figures 2.11 and 2.12 compare the experimental recovery (%) and grade (%BPL) with those predicted by the hybrid model, respectively, for the fine feed size distribution. As seen from these figures, the hybrid model fails to successfully predict both recovery and grade. This is attributed to the fact that fine feed has a very wide size distribution (35X150 Tyler mesh size) and only the overall recovery and grade were measured experimentally. It is therefore necessary to utilize narrow ranges of feed size and to analyze for recovery and grade according to each size range instead of just one recovery and grade for the entire particle size distribution. This was implemented for the unsized feed size which has even a wider size distribution (14X150 Tyler mesh). Figures 2.13 and 2.14 compare the experimental recovery (%) and grade (%BPL) predicted by the hybrid model, respectively, for the unsized feed after it has been sized and grade and recovery was determined for each size. As can be seen from these figures, the hybrid model successfully predicts both recovery and grade.

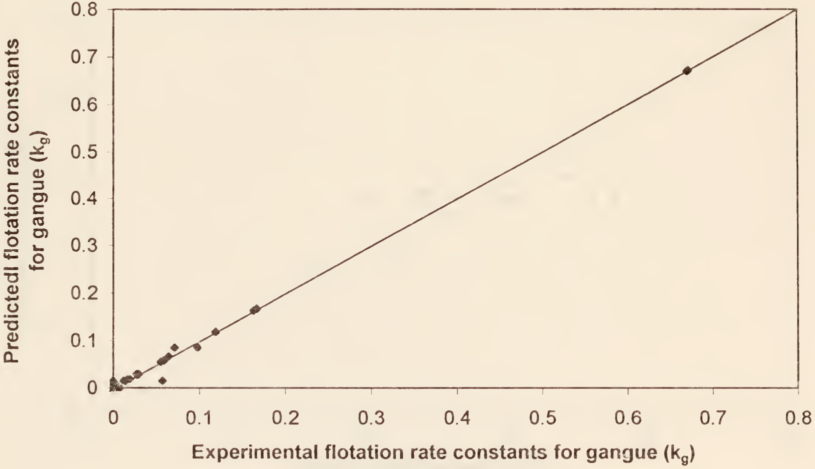

Figure 2.7 Performance of NNII: Model versus experimental flotation rate constant for gangue (k_g)

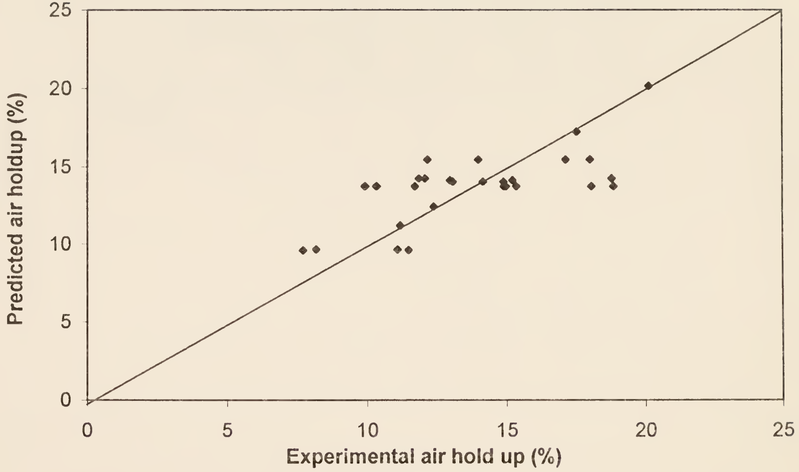

Figure 2.8: Performance of NNIII: Model versus experimental air holdup for frother CP-100

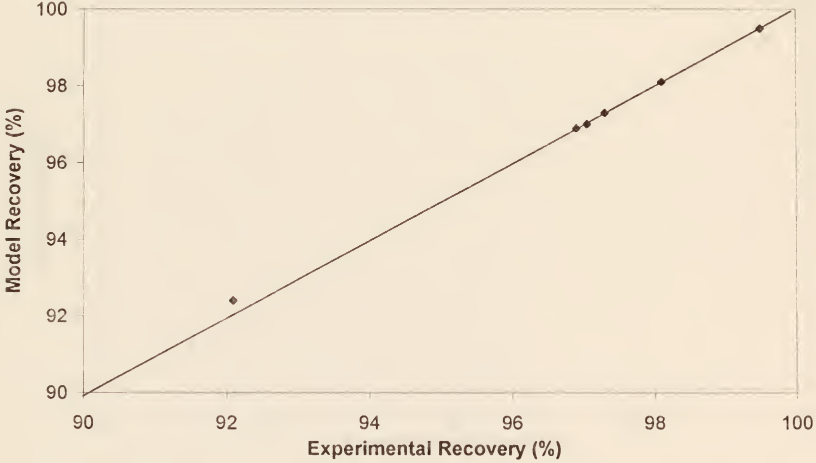

Figure 2.9 Performance of the overall hybrid model: Predicted versus experimental recovery (%) for coarse feed size distribution

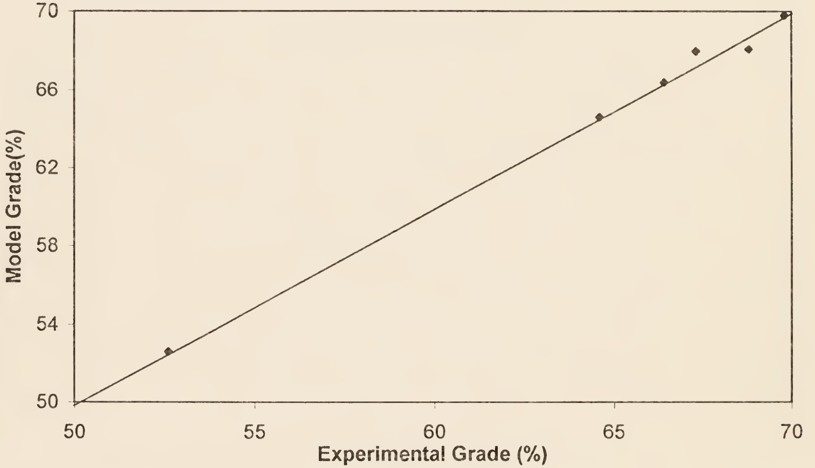

Figure 2.10: Performance of the overall hybrid model: Predicted versus experimental grade (%BPL) for coarse feed size distribution

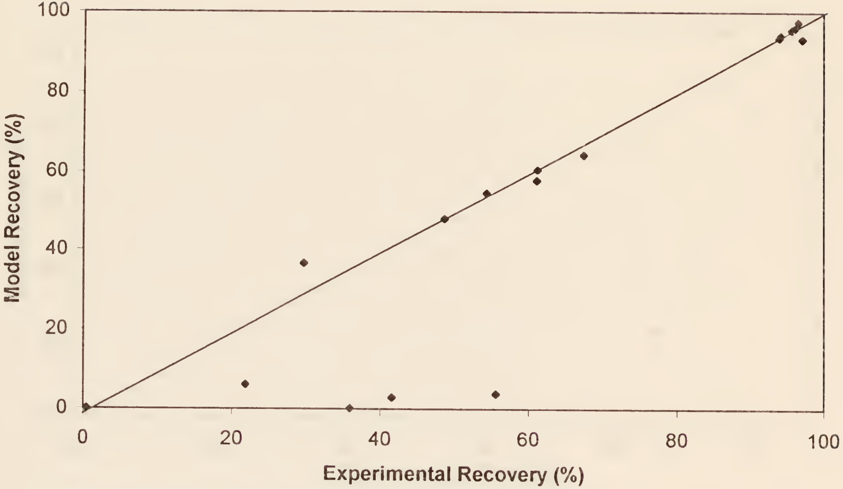

Figure 2.11: Performance of the overall hybrid model: Predicted versus experimental recovery (%) for fine feed size distribution

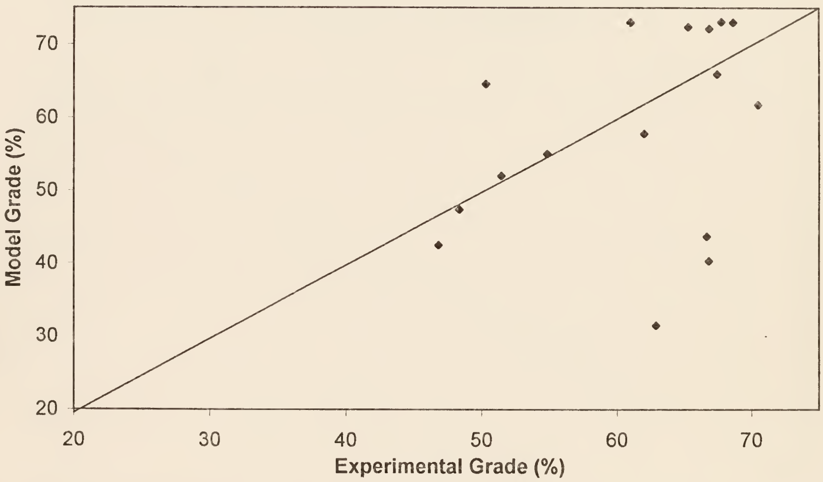

Figure 2.12: Performance of the overall hybrid model: Predicted versus experimental grade (%BPL) for fine feed size distribution

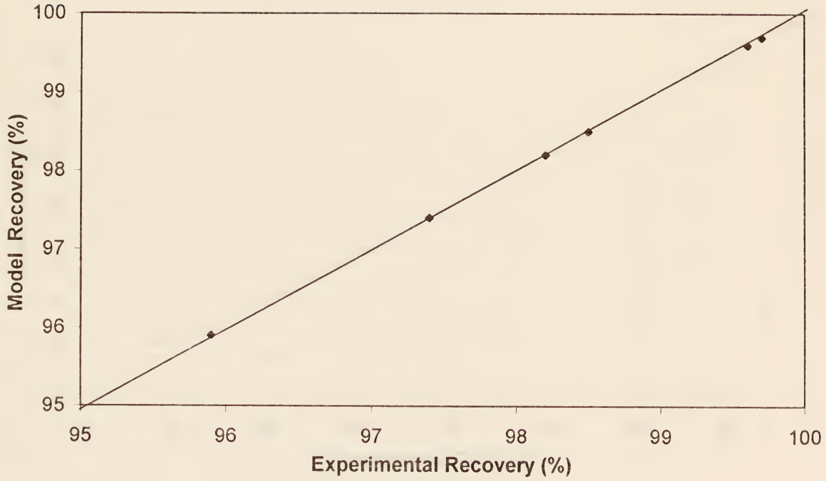

Figure 2.13: Performance of the overall hybrid model: Predicted versus experimental recovery (%) for the unsized feed after it has been sized.

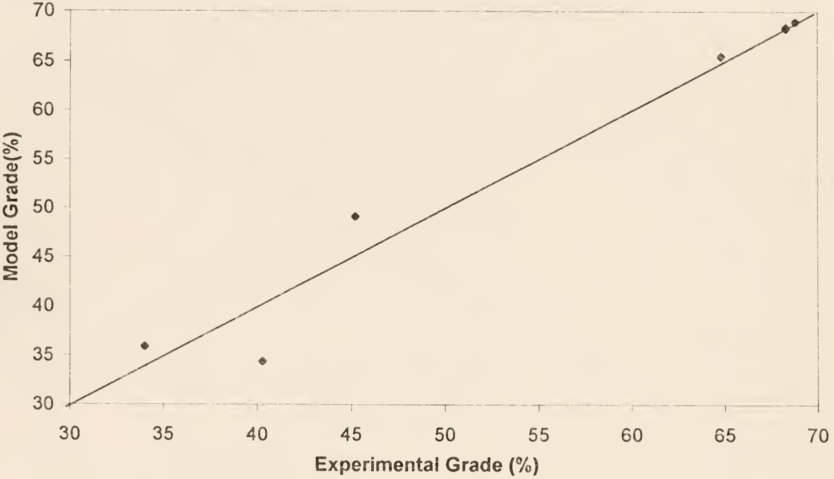

Figure 2.14: Performance of the overall hybrid model: Predicted versus experimental grade (%BPL) for the unsized feed after it has been sized.

2.6 Conclusions

In this work, we have demonstrated that a one-phase first-principles model can effectively be coupled with the artificial neural networks for predicting the grade and recovery of a phosphate flotation column with negative bias. Artificial neural networks are used to predict the flotation rate constants and air holdup. Experimental data from a lab-scale column were used to train the neural networks. The hybrid model successfully predicts the effects of particle size, superficial air velocity, frother concentration, collector concentration, extender concentration, and pH.

CHAPTER 3
TWO-LEVEL HYBRID MODEL

A new model for phosphate column flotation is presented which relates the effects of operating variables such as frother concentration and air flow rate on column performance. This is a hybrid model that combines a first-principles model with artificial neural networks. The first-principles model is obtained from material balances on both phosphate particles and gangue (undesired material containing mostly silica). First order rates of net attachment are assumed for both. Artificial neural networks relate the attachment rate constants to the operating variables. Experiments were conducted in a 6" diameter laboratory column to provide data for neural network training and model validation. The model is shown to successfully predict the effects of frother concentration, particle size, air flow rate, and bubble diameter on grade and recovery.

3.1 Introduction

Flotation is a process in which air bubbles are used to separate a hydrophobic from a hydrophilic species. The majority of the hydrophobic material gets attached to the bubbles and leaves with the froth from the top of a cell or column separator, while the hydrophilic material leaves from the bottom. This process is commonly used in the minerals industry, including the phosphate industry, in which case the phosphate containing rock (frankolite or apatite) is to be separated from gangue (mostly silica). Flotation is also used to remove oil from wastewater and to remove ink from paper pulp.

In anionic phosphate flotation the mineral is first treated with fatty acid collector and fuel oil extender. At proper concentrations these mostly adsorb on the phosphate-containing particles rendering them hydrophobic. Then the phosphate-containing particles are separated from gangue via the flotation process. The majority of the phosphate plants employ mechanical cells. However, column flotation has simpler operation and provides superior grade/recovery performance. For these reasons column flotation is gaining increasing acceptance for the processing and beneficiation of phosphate ores.

Column flotation is frequently employed for the recovery of other minerals (e.g., coal, copper, nickel, gold). In such applications the column can be divided into three zones: an upper froth zone, a lower collection zone, and an intermediate interface zone. An additional "wash water" stream is usually added from the top of the column. Phosphate flotation deals with considerably larger particles of size 0.1-1 mm. As a result, instead of wash water from the top, elutriation water from the bottom is added. Furthermore, columns are typically operated with negligible froth and interface zones. This considerably simplifies the modeling effort, as the only the collection zone needs to be accounted for.

Particle transport in the collection zone is usually modeled as axial convection coupled with axial dispersion. The Peclet number (Pe), or its inverse, the dispersion number, governs the degree of mixing. Most models only consider the slurry phase (Finch and Dobby, 1990; Luttrell and Yoon, 1993), in which case particle collection is viewed as a first order net attachment rate process. A model that considers both slurry and air phase was developed by Sastry and Loftus (1988). In this case particle attachment and detachment are modeled separately with first order rates. Luttrell and Yoon (1993) used a probabilistic approach to relate the particle net attachment rate constant to some operating variables (e.g., air flow rate). However, their approach involves empirical parameters and it cannot be used to predict the effect of certain operating variables such as the frother and collector concentrations.

In this work, we use neural networks to determine the dependence of the phosphate and gangue flotation rate constants on the operating variables. Artificial neural networks have the ability to approximate any smooth nonlinear multivariable function arbitrarily well (Hornik *et al.*, 1989). This approach can be used to determine the dependence of the performance of a flotation column (i.e., grade and recovery) on any operational variable. We demonstrate it in this work by developing a hybrid model that predicts the effect of frother concentration, air flow rate, feed rate and loading, elutriation flow rate, tailings flow rate, and particle size distribution.

The idea of developing a hybrid model by combining a first-principles model (FPM) with artificial neural networks (ANNs) is not new. Johansen and Foss (1992) and Su *et al.* (1992) proposed parallel structures where the output of the hybrid model is a weighted sum of the first-principles and ANN models. Kramer *et al.* (1992) proposed a parallel arrangement of a default model (which could be a first principles model) and a radial basis function ANN. An alternative approach is to combine ANNs with a FPM in a serial fashion, by using the ANNs to develop expressions for the FPM parameters or rate expressions. Psichogios and Ungar (1992a, 1992b) proposed this scheme for parameters that are functions of the state variables and manipulated inputs, and trained the neural networks (i.e. determined the neural network parameters) on the error of the output of the first-principles model. A similar approach was followed by Reuter *et al.* (1993) to model metallurgy and mineral processes. Liu *et al.* (1995) developed a hybrid model for a periodic wastewater treatment process by using ANNs for the bio-kinetic rates of a first-principles model. The Psichogios and Ungar (1992a, 1992b) approach was used by Cubillo et al. (1996) to model particulate drying processes, and by Cubillo and Lima (1997) to develop a hybrid model for a rougher flotation circuit. Thompson and Kramer (1994) combined the parallel and serial hybrid modeling approaches.

As in the Psichogios and Ungar (1992a, 1992b) approach, the hybrid model presented here uses backpropagation ANNs for certain parameters of a FPM. However, instead of training these ANNs on the errors of the measured outputs of the FPM (grade and recovery), it inverts the FPM for each set of measurements to calculate corresponding parameter values, and trains the ANNs on the errors of the calculated parameter values. Another innovation of the present hybrid model is that it involves two levels of neural networks. This structure has the advantage that if certain factors that affect the process like the type of frother or air sparger used are changed, only the top level neural networks need to be retrained. These only require experimental data that can be easily obtained with short experiments that do not involve rock, and the large database of past grades and recoveries is still valid and does not need to be replaced. Finally, in contrast to the above referenced works, the hybrid model presented here is developed with experimental data instead of simulated data.

The next section presents the first-principles model. The subsequent section deals with the calculation of model parameters from measured outputs. This is followed by a discussion of the artificial neural networks and their integration with the first-principles model to develop a hybrid model. The fourth section describes the experimental setup, materials used, experimental procedure, and the methodology used to train the neural networks. The final section presents results and compares the model predictions of grade and recovery to experimentally measured grade and recovery.

3.2 First-Principles Model

The FPM is obtained from material balances on both phosphate and gangue. It neglects radial dispersion and changes in the air holdup. Following Luttrell and Yoon (1993) the particle to bubble attachment and detachment rates are combined in one net attachment rate, and this rate is assumed to be first order with respect to particle concentration in the slurry.

The model subdivides the column into n layers as shown in Figure 3.1. Feed containing both the desired (phosphate) and undesired (gangue) particles enters in a slurry in layer k. An additional inlet stream is the elutriation water that enters in the bottom of the column (layer n). Most of that flow is due to water that enters with the air sparger, as most of the popular spargers are two-phase and introduce a considerable amount of water. There are two outlet streams: the tailings stream through the bottom of the column (layer n) that contains mostly gangue, and the product (concentrate) stream that leaves from the top of the column.

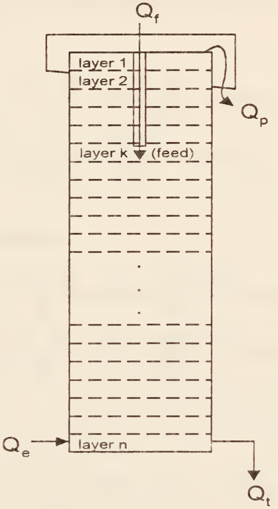

Figure 3.1: Schematic diagram of column for phosphate flotation.

The particles are subdivided into size ranges according to the standard Tyler mesh screens. Particles of a certain mesh are considered to have diameter the geometric mean of the lower and upper limits. As the attachment rate constants and particle slip velocities depend on particle size, a separate material balance is written for each mesh size. Material balances at each layer yield the following equations for the phosphate particles:

Layer 1 (top)

$$\frac{dC^j_{p_1}}{dt}=\begin{cases}\left(\dfrac{U_p}{1-\varepsilon_g}-U^j_{sl}\right)\dfrac{C^j_{p_2}-C^j_{p_1}}{\Delta z}+D\,\dfrac{C^j_{p_2}-C^j_{p_1}}{\Delta z^2}-k_p(d^j_p)C^j_{p_1} & \text{if } U^j_{sl}\le\dfrac{U_p}{1-\varepsilon_g}\\ -\left(U^j_{sl}-\dfrac{U_p}{1-\varepsilon_g}\right)\dfrac{C^j_{p_1}}{\Delta z}+D\,\dfrac{C^j_{p_2}-C^j_{p_1}}{\Delta z^2}-k_p(d^j_p)C^j_{p_1} & \text{if } U^j_{sl}>\dfrac{U_p}{1-\varepsilon_g}\end{cases}\tag{3.1}$$

Layer 2 to k-1; k = feed layer

$$\frac{dC^j_{p_i}}{dt}=\begin{cases}\left(\dfrac{U_p}{1-\varepsilon_g}-U^j_{sl}\right)\dfrac{C^j_{p_{i+1}}-C^j_{p_i}}{\Delta z}+D\,\dfrac{C^j_{p_{i+1}}-2C^j_{p_i}+C^j_{p_{i-1}}}{\Delta z^2}-k_p(d^j_p)C^j_{p_i} & \text{if } U^j_{sl}\le\dfrac{U_p}{1-\varepsilon_g}\\ \left(U^j_{sl}-\dfrac{U_p}{1-\varepsilon_g}\right)\dfrac{C^j_{p_{i-1}}-C^j_{p_i}}{\Delta z}+D\,\dfrac{C^j_{p_{i+1}}-2C^j_{p_i}+C^j_{p_{i-1}}}{\Delta z^2}-k_p(d^j_p)C^j_{p_i} & \text{if } U^j_{sl}>\dfrac{U_p}{1-\varepsilon_g}\end{cases}\tag{3.2}$$

Feed Layer = k

$$\frac{dC^j_{p_k}}{dt}=\begin{cases}\dfrac{(Q_f/A_c)C^j_f-(\frac{U_p}{1-\varepsilon_g}-U^j_{sl})C^j_{p_k}-(\frac{U_t}{1-\varepsilon_g}+U^j_{sl})C^j_{p_k}}{\Delta z}+D\,\dfrac{C^j_{p_{k+1}}-2C^j_{p_k}+C^j_{p_{k-1}}}{\Delta z^2}-k_p(d^j_p)C^j_{p_k}\\ \qquad \text{if } U^j_{sl}\le\dfrac{U_p}{1-\varepsilon_g}\\ \dfrac{(Q_f/A_c)C^j_f+(U^j_{sl}-\frac{U_p}{1-\varepsilon_g})C^j_{p_{k-1}}-(\frac{U_t}{1-\varepsilon_g}+U^j_{sl})C^j_{p_k}}{\Delta z}+D\,\dfrac{C^j_{p_{k+1}}-2C^j_{p_k}+C^j_{p_{k-1}}}{\Delta z^2}-k_p(d^j_p)C^j_{p_k}\\ \qquad \text{if } U^j_{sl}>\dfrac{U_p}{1-\varepsilon_g}\end{cases}\tag{3.3}$$

Layer k+1 to n-1

$$\frac{dC^j_{p_i}}{dt}=\left(\frac{U_t}{1-\varepsilon_g}+U^j_{sl}\right)\frac{C^j_{p_{i-1}}-C^j_{p_i}}{\Delta z}+D\,\frac{C^j_{p_{i+1}}-2C^j_{p_i}+C^j_{p_{i-1}}}{\Delta z^2}-k_p(d^j_p)C^j_{p_i} \tag{3.4}$$

Layer n (bottom)

$$\frac{dC^j_{p_n}}{dt}=\begin{cases}\dfrac{\left(\dfrac{U_t}{1-\varepsilon_g}+U^j_{sl}\right)C^j_{p_{n-1}}-\left(\dfrac{Q_t}{(1-\varepsilon_g)A_c}+U^j_{sl}\right)C^j_{p_n}}{\Delta z}+D\dfrac{C^j_{p_{n-1}}-C^j_{p_n}}{\Delta z^2}-k_p(d^j_p)C^j_{p_n} & \text{if } U^j_{sl}\geq-\dfrac{U_t}{1-\varepsilon_g}\\[2ex] \dfrac{\left(\dfrac{U_t}{1-\varepsilon_g}+U^j_{sl}\right)C^j_{p_n}-\left(\dfrac{Q_t}{(1-\varepsilon_g)A_c}+U^j_{sl}\right)C^j_{p_n}}{\Delta z}+D\dfrac{C^j_{p_{n-1}}-C^j_{p_n}}{\Delta z^2}-k_p(d^j_p)C^j_{p_n} & \text{if } U^j_{sl}<-\dfrac{U_t}{1-\varepsilon_g}\end{cases} \tag{3.5}$$

where

A_c = Cross-sectional area of the column

C^j_f = Phosphate feed concentration of j^th mesh size particles

$C^j_{p_i}$ = Phosphate concentration of j^th mesh size particles in the i^th layer

Q_f = Feed volumetric flow rate

Q_t = Tailings volumetric flow rate

Q_e = Elutriation volumetric flow rate

Q_p = Product volumetric flow rate

U_p = Superficial liquid velocity above the feed point

= Q_p/A_c

U_t = Superficial liquid velocity below the feed point

= $(Q_t-Q_e)/A_c$

U^j_{sl} = Slip velocity of j^th mesh size particles

ε_g = Air holdup

$k_p(d^j_p)$ = Flotation rate constant for phosphate for j^th mesh size particles

The slip velocity is calculated using the expression of Villeneuve *et al.* (1996):

$$U_{sl}^{J} = \frac{g\, d_p^{J^2} (\rho_s - \rho_l)(1-\phi_s)^{2.7}}{18\mu_l (1 + 0.15 R_{ep}^{J^{0.687}})} \tag{3.6}$$

where the particle Reynolds number is defined as

$$R_{ep}^{J} = \frac{d_p^J\, U_{sl}^J \rho_s (1-\phi_s)}{\mu_l} \tag{3.7}$$

where

g = Acceleration due to gravity (m/s^2)

μ_l = Water viscosity (kg/ms)

ρ_l = Water density (kg/m^3)

ρ_s = Solid density (kg/m^3)

ϕ_s = Volume fraction of solids in slurry

d_p^J = Particle diameter (m)

As the right hand side of Equation 3.7 is a function of U_{sl}^J, the slip velocity is obtained by solving Equations 3.6 and 3.7 iteratively as described in Chapter 2.

The axial dispersion coefficient is calculated by a modification of the Finch and Dobby (1990) expression:

$$D = 0.063\,(1-\varepsilon_g)\, d_c \left(\frac{J_g}{1.6}\right)^{0.3} \tag{3.8}$$

where

d_c = column diameter (m)

J_g = superficial air velocity (cm/s)

Equations analogous to 3.1-3.8 are valid for the gangue particles, but with a considerably lesser effective flotation rate constant $k_g(d_p^j)$.

In the limit as $\Delta z \to 0$ the above difference equations become

$$\frac{\partial C_p^j}{\partial t} = \left(\frac{U_p}{1-\varepsilon_g} - U_{sl}^j \right) \frac{\partial C_p^j}{\partial z} + D \frac{\partial^2 C_p^j}{\partial z^2} - k_p(d_p^j) C_p^j \tag{3.9}$$

for the section above the feed and

$$\frac{\partial C_p^j}{\partial t} = \left(\frac{U_t}{1-\varepsilon_g} + U_{sl}^j \right) \frac{\partial C_p^j}{\partial z} + D \frac{\partial^2 C_p^j}{\partial z^2} - k_p(d_p^j) C_p^j \tag{3.10}$$

for the section below the feed.

Recovery (%) is defined as the ratio of the weight of the phosphate in the concentrate stream to the weight of the phosphate in the feed. The recovery for phosphate particles of the j^{th} mesh size can be expressed in terms of the feed and tailings flow rates and concentrations as

$$R_p^j = \left(\frac{Q_f C_f^j - \left[Q_t + A_c(1-\varepsilon_g) U_{sl}^j \right] C_{p_n}^j}{Q_f C_f^j} \right) * 100 \tag{3.11}$$

Grade, a measure of the quality of the product, is defined as the ratio of the weight of the phosphate to the total weight recovered in the concentrate stream. Grade is usually reported as % Bone Phosphate of Lime (%BPL) which is the equivalent grams of tricalcium phosphate, $Ca_3(PO_4)_2$, in 100 g of sample. For the typical Florida rock, mineral that contains no gangue is 73.3 %BPL. Grade can be obtained as the ratio of phosphate to the sum of phosphate and gangue in the product:

$$G^j = \left(\frac{Q_f C_f^j - \left[Q_t + A_c(1-\varepsilon_g)U_{sl}^j\right] C_{p_n}^j}{(Q_f C_f^j - [Q_t + A_c(1-\varepsilon_g)U_{sl}^j]C_{p_n}^j) + (Q_f C_{f_g}^j - [Q_t + A_c(1-\varepsilon_g)U_{sl}^j]C_{g_n}^j)} \right) * 73.3$$

(3.12)

where $C_{g_n}^j$ is gangue concentration of jth particle size in the nth layer and $C_{f_g}^j$ is the gangue feed concentration of jth particle size. The algorithm for solving the first-principles model is given in Appendix B.

3.3 Calculation of Model Parameters

Since air-holdup ε_g is measured experimentally, the above FPM has only two unmeasured model parameters for each particle size, namely, the flotation rate constants for phosphate (k_p) and for gangue (k_g). The experimental analysis usually available in industrial flotation columns is in terms of grade and recovery of phosphate. Let W_{fp}^j denote the weight of jth size phosphate particles in the feed, W_{fg}^j the weight of jth size gangue in the feed, and W_g^j the weight of jth size gangue in the product. The grade of feed is then

$$G_f^j = 73.3 \frac{W_{fp}^j}{W_{fp}^j + W_{fg}^j} \tag{3.13}$$

and G^j is given by an analogous expression. The recovery of gangue can be readily calculated from measurements of grade and recovery of phosphate using the following relationship:

$$R_g^J = \frac{W_g^J}{W_{fg}^j} = \frac{R_p^J G_f^J (73.3 - G^J)}{G^J (73.3 - G_f^J)} \tag{3.14}$$

In some cases direct measurements of the majority of gangue as acid insolubles may be available. Then more reliable estimates of R_g^J can be obtained by averaging values calculated from measurements of acid insolubles with values calculated from Equation 3.14. This was done in this work.

From the FPM equations follows that the recovery of phosphate depends only on k_p, while the recovery of gangue depends only on k_g. This can be exploited to easily invert the steady-state version of the model to determine from experimental measurements of R_p^j and G^J corresponding k_p and k_g. As shown in Figure 2.1, this is accomplished with one-dimensional searches. The search for k_p is initialized with two values that yield errors in the corresponding recovery R_p^J of opposite sign. Since typically $0 \leq k_p \leq 10$ min^{-1} the values of 0 and 100 min^{-1} are used. Then the method of false position (Chapra and Canale, 1988) is used to iterate until the magnitude of the error in R_p^J drops to less than 10^{-3}. It is possible that the calculated recovery has a higher value than the experimental even for $k_p = 0$. In these cases k_p is set equal to zero. The above procedure is also used to determine k_g, except that the high initial value is set to 10 min^{-1}. Recovery for both phosphate and gangue increases monotonically with respective flotation rate constants as discussed in Chapter 2.

3.4 The Hybrid Model

The main factors affecting the air hold up ε_g are the superficial air velocity J_g and the frother concentration $C_{frother}$. Several factors affect the flotation rate constants, k_p and k_g, including particle diameter, superficial air velocity, frother concentration, collector concentration, extender concentration, and pH. In this study we have conducted experiments varying particle size, frother type, frother concentration, and superficial air velocity, and develop a hybrid model that portrays the effect of these factors on the performance of the column. The hybrid model utilizes backpropagation ANNs (Rumelhart and McClelland 1986) to predict the values of the parameters ε_g, k_p, and k_g.

The straightforward approach is to develop an ANN for each of the three parameters. The inputs to the ANNs that predict k_p and k_g would be d_p, J_g, and $C_{frother}$, while the inputs of the ANN that predicts ε_g would be J_g, and $C_{frother}$. Each of the ANNs in this structure would then depend on the frother and sparger used. A change in type of frother would mean that the previously trained ANNs are no longer applicable and would necessitate collection of a new set of training data and retraining of the networks. As changes in frother or sparger are not uncommon, this is a disadvantage.

The main reason J_g and $C_{frother}$, as well as the type of frother and sparger, affect the flotation rate constants, is because they significantly affect the bubble size. An alternative hybrid model architecture is shown in Figure 3.2. The neural networks are structured in two levels. The first level consists of the ANNs for predicting k_p (NNI) and

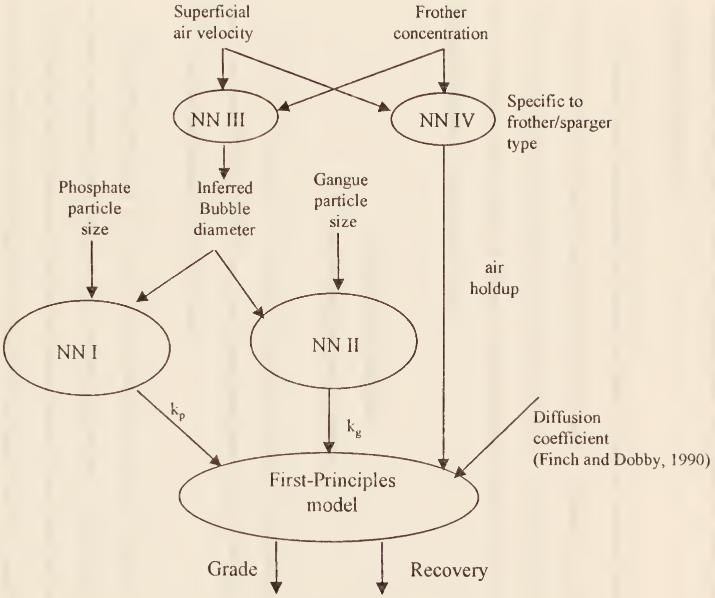

Figure 3.2 : Overall structure of the hybrid model

k_g (NNII) and receives as an input the inferred bubble size. This is the output of one of the ANNs of the second (top) level, NNIII. The second level also includes NNIV, which predicts air holdup. The advantage of this structure is that NNI and NNII are independent of the type of frother and sparger used, and therefore would not need retraining if these change.

As bubble size is not measured in industry, we infer it from the two-phase (air/water) air holdup, J_g, and U_t using the well-known Drift-flux analysis (Yianatos *et al.*, 1988). The output required to train NNIV is the (two-phase) air holdup. Air holdup is relatively easy to obtain, so after a change of frother or sparger the hybrid model of Figure 3.2 can become functional in a short interval of time.

3.5 Materials and Methods

3.5.1 Experimental Setup and Procedures

Two types of experiments were conducted: two-phase (air/water) experiments to train neural networks NNIII and NNIV, and three-phase experiments to train NNI and NNII and to test the performance of the hybrid model.

The experimental setup for the three-phase experiments is shown in Figure 2.5. It included an agitated tank (conditioner) for reagentizing the feed, a screw feeder for controlling the rate of reagentized feed, and a flotation column. The agitated tank was 45 cm in diameter and 75 cm high and was equipped with an impeller with two axial blades (each 28 cm diameter). The impeller had about 3.8 cm clearance from the bottom of the tank and its rotation speed was fixed at 465 rpm. The feeder with a 2.5 cm diameter screw delivered the conditioned phosphate materials to the column. The feed rate was controlled by adjusting the screw rotation speed. The flotation column was constructed of plexiglass and had 14.5 cm diameter and 1.82 m height. The feeding point was located at 30 cm from the column top. The discharge flow rate was controlled by a discharge valve and an adjustable speed pump. Three flowmeters were used to monitor the flow rates for air, frother solution, and elutriation water.

Phosphate feed (14X150 Tyler mesh) from Cargill was used as the feed material. For each run, 50 kg of feed were introduced to the pre-treatment tank and water was added to obtain 72 % solids concentration by weight. The tank was then agitated for 10 seconds. 10 % soda ash solution was added to the pulp to reach pH of about 9.4 and the slurry was agitated for another 10 seconds. Subsequently, a mixture of fatty acids (a mixture of oleic, palmetic, and linoleic acid obtained from Westvaco) and fuel oil (No. 5 obtained from PCS Phosphates) with a ratio of 1:1 by weight was added to the pulp and the slurry continued to be mixed. The total conditioning time was 3 minutes. The conditioned feed material (without its conditioning water) was subsequently loaded to the feeder bin located at the top of the column.

Four frothers were used, two commonly employed in industry, F-507 (a mixed polyglycol by Oreprep) and CP-100 (a sodium alkyl ether sulfate by Westvaco), and two experimental, F-579 (also a mixed polyglycol by Oreprep) and OB-535 (by O'Brien). Frother-containing water and air were first introduced into the column through the sparger (an eductor) at a fixed water flow rate and frother concentration (0 – 30 ppm), and the superficial air velocity ranged from 0.24 – 0.94 cm/s. Then the discharge valve and pump were adjusted to get the desired underflow and overflow rates. Air holdup was measured using a differential pressure gauge. After the water/air system reached steady state, the screw feeder was started. To achieve steady feed rate to the column, water was added to the screw feeder at the rate that reduced the solids concentration to approximately 66% by weight. The column was run for a period of three minutes with phosphate feed prior to sampling. Timed samples of tailings and concentrates were taken. The collected product samples, as well as feed samples, were dried, sieved using Tyler meshes, weighed and analyzed for %BPL following the procedure recommended by the Association of Florida Phosphate Chemists (AFPC Analytical Methods, 1980). In addition, gangue content (as % acid insolubles) of the feed, tailings, and concentrate streams was measured (AFPC Analytical Methods, 1980). These measurements were then used to calculate recovery of acid insolubles. Subsequently these values were averaged with the values obtained from Equation 3.14 to obtain the R_g^j used to determine the flotation rate constants for gangue.

The two-phase experiments were identical to the three-phase experiments, except that no solid feed was introduced to the column and the experiments were terminated when the water/air system reached steady state.

3.5.2 Neural Network Structure and Training

NNI, NNII, NNIII, and NNIV of Figure 3.2 were feedforward backpropagation artificial neural networks with a single layer of hidden nodes between the input and output layers and a unit bias connected to both the hidden and the output layers. Inputs and outputs were scaled from 0 to 1. The hidden and output layer nodes employed logistic activation functions (Hertz *et al.*, 1992).

For each of the four frothers investigated, 28 two-phase experiments were conducted (full factorial design with 7 frother concentrations and 4 superficial air velocities). These were used to train (19 data points) and to validate (9 data points) the top level neural networks (NNIII and NNIV), a different pair for each frother. Three-phase runs yielded 28 experimental grades and recoveries, which were used to train (19 data points) and to validate (9 data points) NNI and NNII. To set the number of nodes in the hidden layer of each network, the number was increased until the sum of the absolute errors of the training and validation outputs started increasing. In this manner an appropriate number of hidden nodes was determined to be three for all the neural networks.

The training process started by initializing all weights randomly to small non-zero values. The random numbers were generated in the range -3.4 to +3.4 with a standard deviation of 1.0 following the procedure recommended by Masters (1993). The optimal weights were determined by combining simulated annealing (Kirkpatrick *et al.* 1983) with the Polak-Ribiere conjugate gradient algorithm (Polak, 1971). Simulated annealing randomly perturbed the independent variables (the weights) and kept track of the best (lowest error) function value for each randomized set of variables. This was repeated several times, each time decreasing the variance of the perturbations with the previous optimum as the mean. Then the conjugate gradient algorithm was used to minimize the mean-squared output error. When the minimum was found, simulated annealing was used to attempt to break out of what may be a local minimum. This alternation was continued until a lower point could not be found. This approach improves the likelihood of convergence to the global optimum.

3.6 Results and Discussion

The performance of the network for predicting bubble diameter (NNIII), the network for predicting air holdup (NNIV), the network for predicting the phosphate flotation rate constant (NNI) and the network for predicting the gangue flotation rate constant (NNII) is shown in Figures 3.3-3.14. Figure 3.3 compares the NNIII output to the inferred bubble diameter using experimental data when the frother was CP-100. The solid circles are for the data used for training while the open squares are for the data used for validation. Figures 3.4, 3.5, and 3.6 show the performance of NNIII when F-507, OB-535, and F-579, respectively, were the frothers.

As these figures show, NNIII successfully predicts the inferred bubble diameter. Figure 3.7 compares the air holdup predicted by NNIV to the experimental values measured by a differential pressure cell when CP-100 was used as the frother. Figures 3.8, 3.9, and 3.10 show the performance of NNIV when F-507, OB-535, and F-579, respectively, were used as frothers. As shown in these figures, NNIV successfully predicts the air holdup for all frothers.

Figures 3.11 and 3.12 show the performance of NNI and NNII, respectively. Figure 3.11 presents the predicted flotation rate constants for phosphate (k_p) against those determined from one-dimensional searches using experimental data. As shown in this figure, NNI does accurately predict low and high values of flotation rate constants.

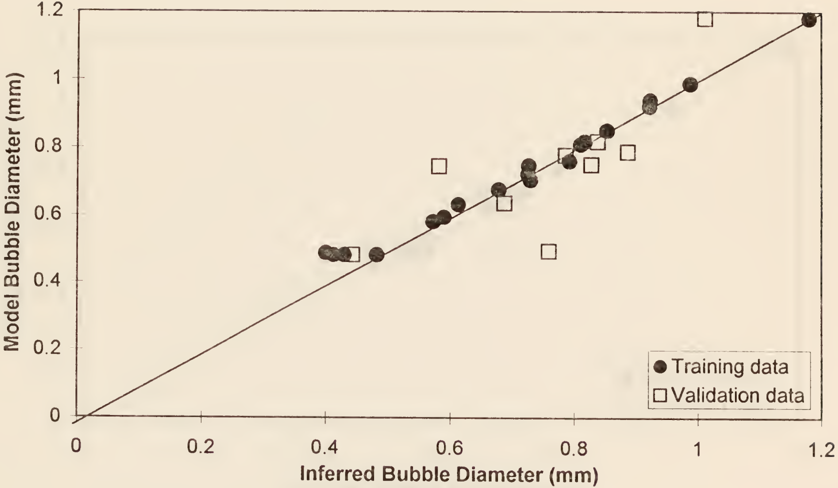

Figure 3.3: Performance of NNIII: Model bubble diameter versus bubble diameter inferred from experimental data when CP-100 was the frother

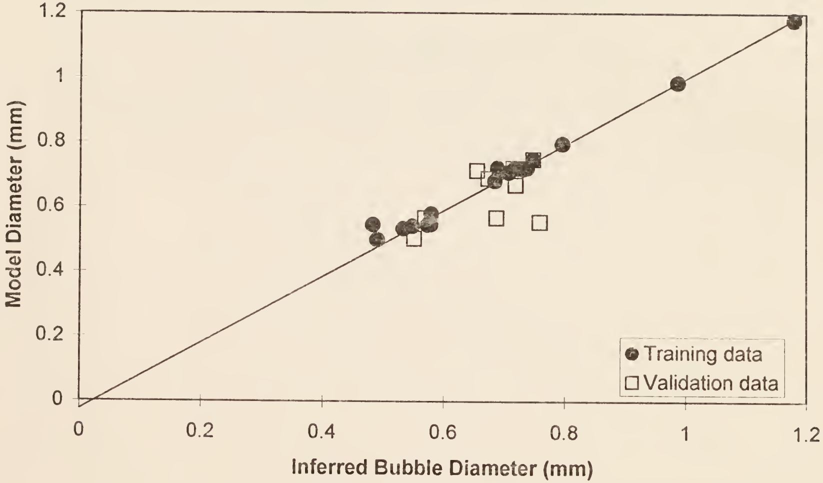

Figure 3.4: Performance of NNIII: Model bubble diameter versus bubble diameter inferred from experimental data when F-507 was the frother

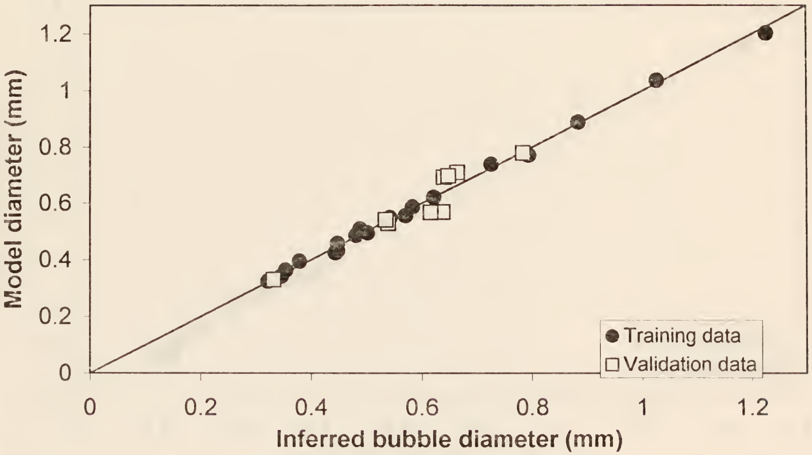

Figure 3.5: Performance of NNIII: Model bubble diameter versus bubble diameter inferred from experimental data when OB-535 was the frother

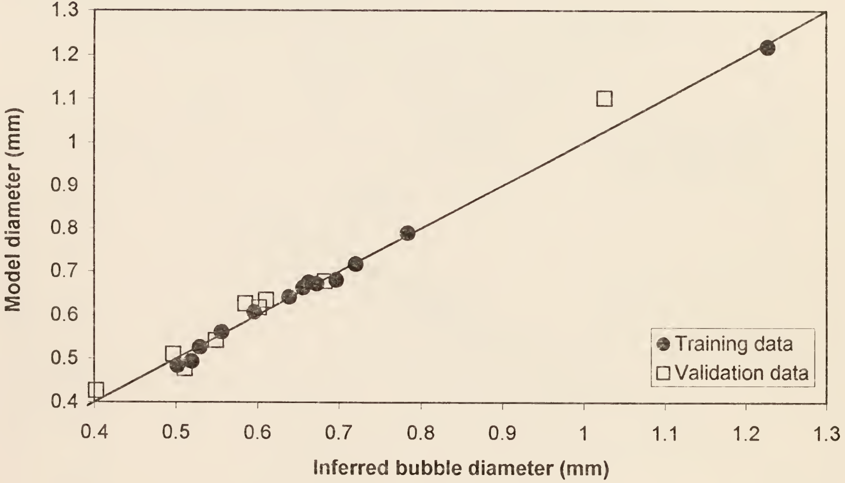

Figure 3.6: Performance of NNIII: Model bubble diameter versus bubble diameter inferred from experimental data when F-579 was the frother

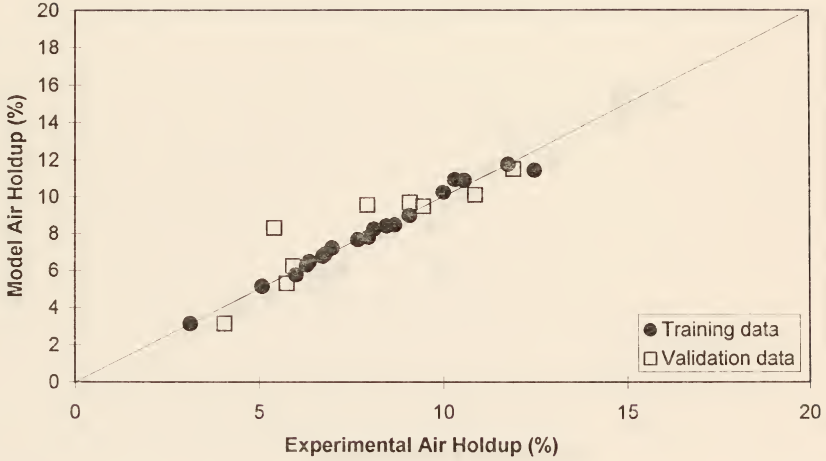

Figure 3.7: Performance of NNIV: Model versus experimental air holdup for frother CP-100

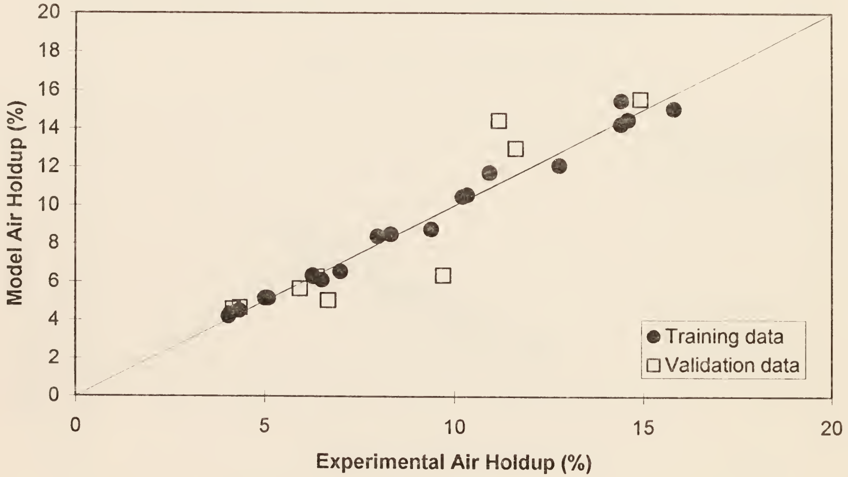

Figure 3.8: Performance of NNIV: Model versus experimental air holdup for frother F-507

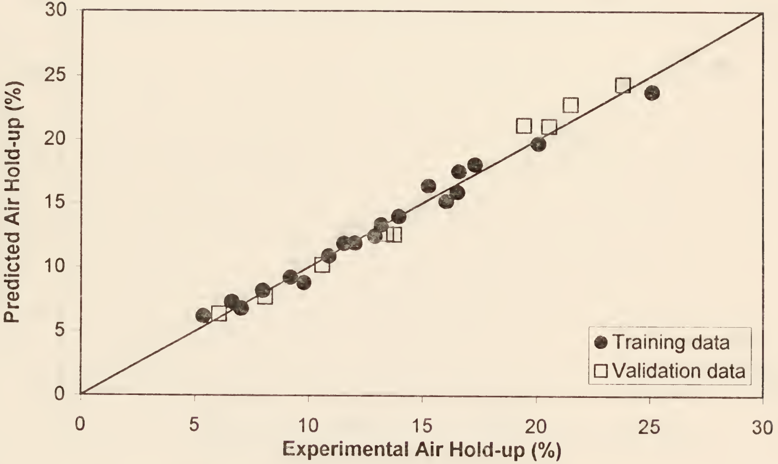

Figure 3.9: Performance of NNIV: Model versus experimental air holdup for frother OB-535

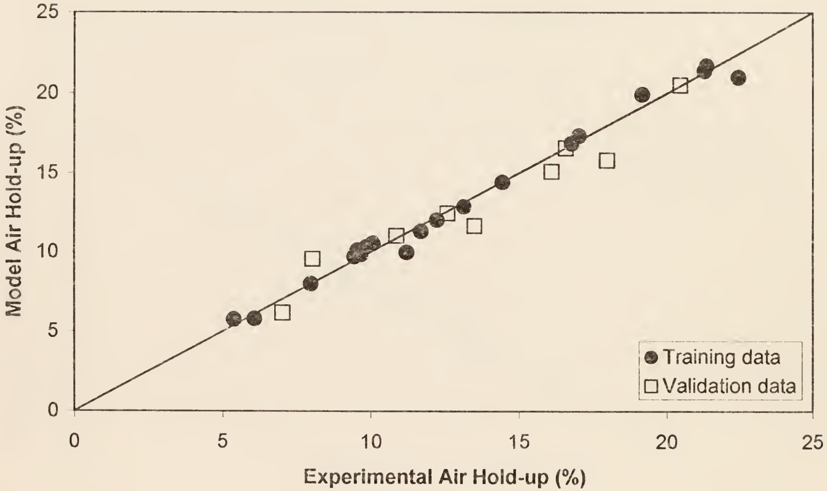

Figure 3.10: Performance of NNIV: Model versus experimental air holdup for frother F-579

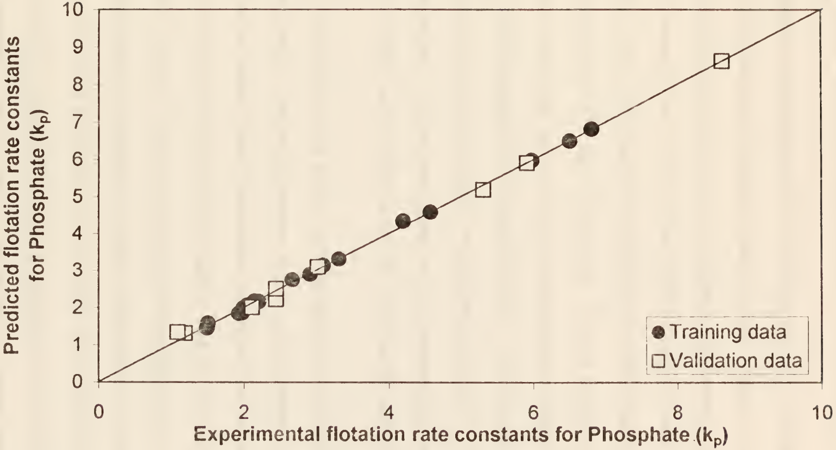

Figure 3.11: Performance of NNI-Model versus experimental flotation rate constant for phosphate (k_p)

Figure 3.12 presents the flotation rate constants for gangue (k_g) predicted using NNII against those determined from experimental data. A very good match is seen.

The hybrid model integrates NNI, NNII, NNIII, and NNIV with the FPM as shown in Figure 3.2. Predictions of the hybrid model are shown in Figures 3.13 and 3.14. Figure 3.13 presents the predicted recovery (%) against the experimental recovery for frother CP-100 (square points), F-507 (circles), OB-535 (triangles), and F-579 (diamonds). Similarly, Figure 3.14 compares the predicted grade (%BPL) against the experimental grade for CP-100, F-507, OB-535, and F-579. It can be seen from these figures that predicted recovery and grade from the hybrid model match closely the experimental values, with the exception of one grade for OB-535. The root mean squared errors in predicted recovery were 0.1%, 0.2%, 1.5%, and 0.4% for CP-100, F-507, OB-535, and F-579, respectively. The root mean squared errors in predicted grade were 3.2 %BPL, 1.5 %BPL, 7.5 %BPL, and 1.5 %BPL for CP-100, F-507, OB-535, and F-579, respectively.

An alternative to the present modeling approach is to develop a pure neural-networks model. This would, however, require a large number of inputs: not only superficial air velocity, frother concentration, and particle size, but also feed flow rate, feed concentration, elutriation flow rate, tailings flow rate, and solids loading. This increase in number of inputs to eight would increase the number of weights (model parameters) needed and therefore the number of three-phase data required for training. Furthermore, as with an in-series hybrid model that uses one level of neural networks, a change in frother or sparger would require generation of a new set of data and retraining of all the networks. The hybrid model presented here with the two levels of neural

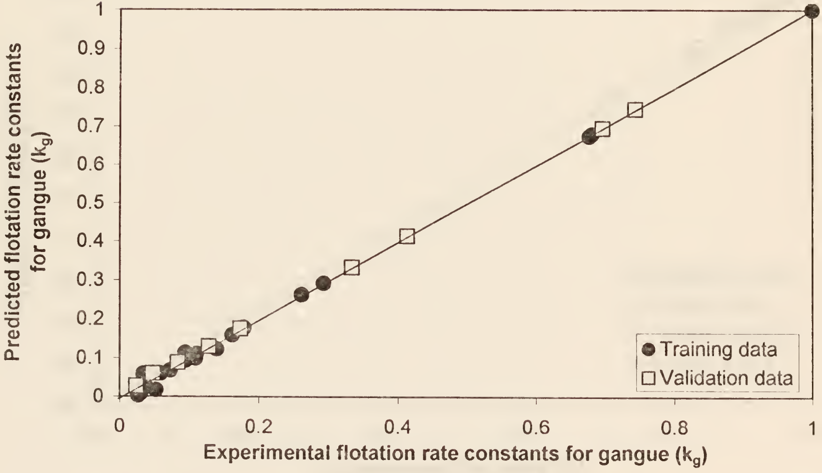

Figure 3.12 Performance of NNII: Model versus experimental flotation rate constant for gangue (k_g)

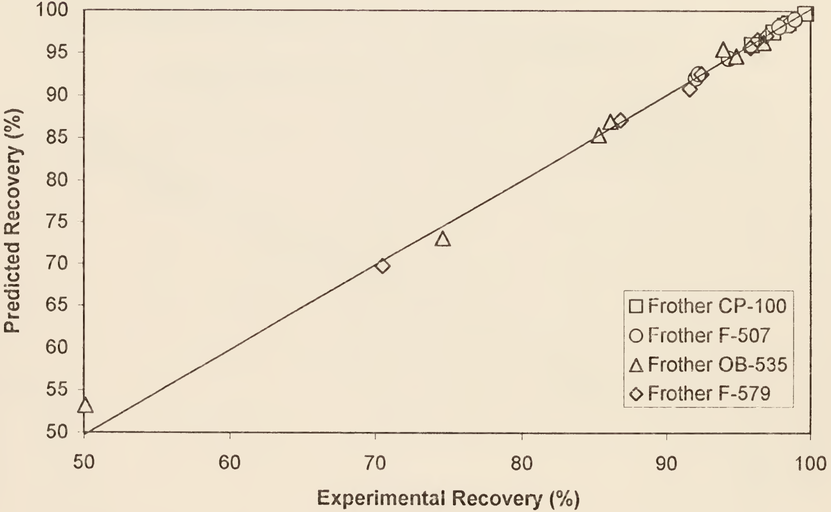

Figure 3.13: Performance of the overall hybrid model: Predicted versus experimental recovery (%) for the four frothers

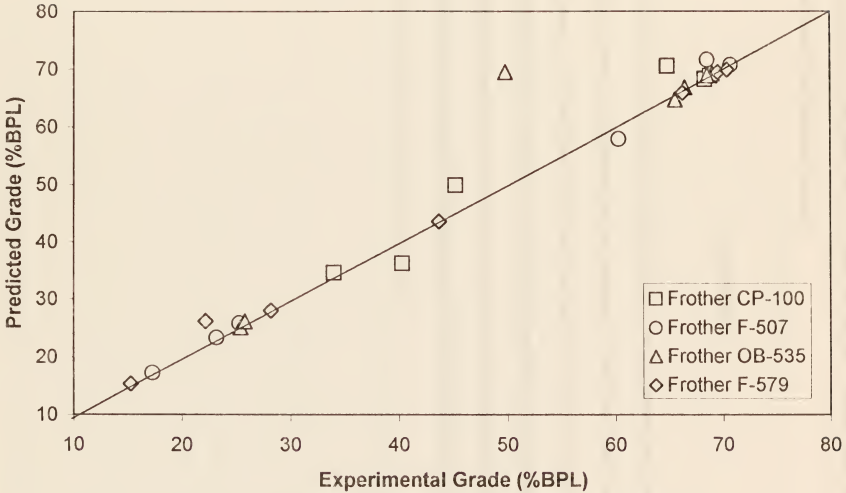

Figure 3.14: Performance of the overall hybrid model: Predicted versus experimental grade (%BPL) for the four frothers

networks involves a relatively low number of inputs in the artificial neural networks, does not require new three-phase data if a frother or sparger is changed, and gives very good predictions of both grade and recovery.

3.7 Conclusions

A hybrid neural network modeling approach was presented and used to model a flotation column for phosphate/gangue separation. This hybrid model is comprised of two parts, a first-principles model and two levels of neural networks that serve as parameter predictors of difficult-to-model process parameters. Experimental data from a laboratory column were used to train and validate the neural networks, and it is shown that the hybrid model captures the dependence of column performance on particle size, frother concentration, and superficial air velocity.

CHAPTER 4
OPTIMIZATION PERFORMANCE MEASURES AND FUTURE WORK

High recovery and grade and low operating cost depend largely on the optimal selection of operating variables. The search of the optimal conditions can considerably benefit by the availability of a model that can relate the operating conditions to the column performance. The hybrid model developed for the flotation column provides a mathematical relationship between the operating variables and column performance. This hybrid model can be combined with an optimization algorithm to determine the optimal operating conditions for the flotation column.

We propose an algorithm that leads to the sequential optimization of a flotation column. This algorithm guides successive changes in the manipulated variables after each experiment to achieve optimal column operating conditions. Selectivity, which combines recovery and grade, can be used as the performance measure of the column. The hybrid model builds a relationship between the process manipulated variables and the performance measure. The optimization algorithm dictates the changes in the manipulated variables between successive runs. At each run manipulated variables are set at their predicted optimal values. After the run is completed, the collected samples should be collected and analyzed for recovery and grade. Then the new input-output data are added to the neural network of the hybrid model and the network should be retrained.

New optimal manipulated variable values are predicted which set the conditions for the subsequent run. This procedure should be repeated until convergence is obtained.

4.1 Performance Measures

The performance of a flotation column is affected by both recovery (%) and grade (%BPL). To guide optimization it is necessary to combine the two outputs (grade and recovery) in a single performance measure. Several performance measures are possible, and some are presented below.

4.1.1 Selectivity

One way to achieve this is to use selectivity as the performance measure. Selectivity is defined as

$$S = \left[\frac{R}{R_b} \Big/ \frac{R_t}{R_{tb}} \right]^{1/2} \tag{4.1}$$

where

R = Recovery of phosphate in the product stream.

R_b = Recovery of gangue in the product stream.

R_t = Recovery(or Rejectability) of phosphate in the tailings stream.

R_{tb} = Recovery(or Rejectability) of gangue in the tailings strea

We developed the following expression that relates selectivity to the recovery and the grade of the product stream

$$S = \left[\frac{\frac{G}{G_f} - G(1-R) - R}{(1-G)(1-R)} \right]^{\frac{1}{2}} \tag{4.2}$$

where

G = Grade (%BPL) of phosphate in the product stream.

G_f = Grade (BPL) of phosphate in the feed.

4.1.2 Separation Efficiency

Separation efficiency is defined as follows:

$$E = R - R_b \tag{4.3}$$

In this case, the efficiency varies between –100 to 100.

4.1.3 Economic Performance Measure

The selectivity function or the separation efficiency does not include any economic input such as cost of the reagents. Therefore an alternate performance measure was developed which includes recovery, grade, and the reagent prices. A scheme for penalizing lower grade rock has been developed. This scheme deducts differential costs, relative to 66% BPL, for transportation and acidulation. The acidulation scheme assumes soluble P_2O_5 losses increase in direct proportion to the amount of phosphogypsum. Thus, the procedure requires an estimate of the quantity of phosphogypsum that is produced.

This performance measure is only applicable to plants, and can not be used with a lab-scale flotation column. The procedure for this scheme is outlined below:

Assumptions

1. The price of rock of 66% BPL = \$22.00
2. Zero insol %BPL = 73.33
3. Transportation cost = \$2.50 per ton.
4. Soluble P_2O_5 losses = 1.00%
5. Insoluble P_2O_5 losses = 6.00%
6. Increase in soluble P_2O_5 losses is proportional to the amount of phosphogypsum produced.

Transportation Penalty

Base case: 66% BPL rock (dry basis)
Freight cost per BPL ton = \$2.5/0.66 = \$3.79

Penalty: $\left(\frac{2.50}{B_L/100}\right) - 3.79$ per BPL ton

$$\textbf{Transportation penalty} = \left(\frac{2.50}{B_L/100} - 3.79\right)\frac{B_L}{100} \text{ per ton}$$

Where, B_L = %BPL when grade < 66%

Acidulation Penalty

Base case: 66% BPL rock (30.21% P_2O_5, $CaO{:}P_2O_5 = 1.49$)

Acid insol $= 100\left(1 - \frac{B_L}{73.33}\right)$

Calculation of the amount of Phosphogypsum:

Phosphogypsum components

Acid insol $= 1 \text{ ton rock} \times \left(1 - \frac{B_L}{73.33}\right)$

Unreacted $= 1 \text{ ton rock} \times \left(\frac{B_L}{73.33}\right) \times 0.06$

Dihydrate $= 1 \text{ ton rock} \times \frac{(B_L/100)}{2.184} \times 1.49 \times (172/56) \times (1 - 0.06)$

Total amount of phosphogypsum = Acid insol + Unreacted + Dihydrate

Soluble P_2O_5 losses $= \$300.0 \times \frac{(\%\text{ soluble } P_2O_5 \text{ losses})/100}{2.184}$ per ton

= \$1.37 per ton

Acidulation Penalty $= \$62.0 \times \frac{(\text{Total amount of phosphogypsum})}{B_L} - 1.37$

Sales value = Price of 66 %BPL rock * $(B_L/66)^{1.5}$

Adjusted sales value = Sales value - Transportation penalty - Acidulation penalty

The adjusted value of the phosphate rock as a function of %BPL is shown in Figure 4.1. Let

Feed solid flow rate = F, ton per year

Product solid flow rate = P, ton per year

Feed grade = G_f, %

Concentrate grade = G, %

Product recovery = R, %

Adjusted sales value of feed = C_f, \$ per ton

Adjusted sales value of product = C_p, \$ per ton

Reagent-i price = C_{ri}, \$/lb

Reagent-i usage = U_i lb/ton feed

The feed flow rate and the product flow rate can be related as:

$$P = F\left(\frac{G_f}{100}\right)\left(\frac{R}{100}\right)\left(\frac{100}{G}\right) \tag{4.4}$$

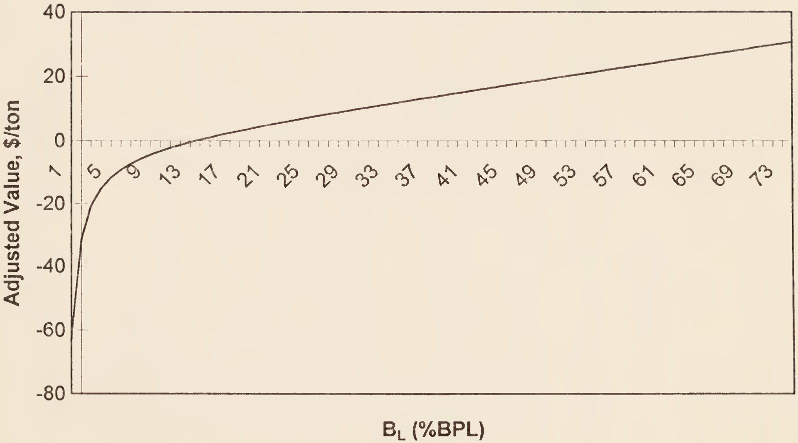

Figure 4.1: Value of phosphate rock as a function of %BPL

$$\textbf{Performance measure} = C_pP - C_fF - F\sum_i U_iC_{ri}\text{, \$/year} \qquad (4.5)$$

4.2 The Optimization Algorithm

The idea behind the sequential optimization is to iterate between experimentation towards the optimum and model identification until the optimum is reached. The procedure is as follows:

(1) Initial experiments are performed and their results are analyzed.

(2) The neural networks are trained and the hybrid model is used to determine the optimal factor values. If these are within the convergence limit of previous experimental values, the procedure stops.

(3) Otherwise, an experiment at the calculated optimal value is performed and analyzed.

(4) The data are added to the neural network training set, and the procedure returns to step (2).

Figure 4.2 shows a more detailed description of the algorithm. After some initialization runs have been completed, the samples are analyzed and the neural networks are trained with the input-output data. Subsequently, using the standard Nelder-Meade algorithm (Himmelblau, 1972), the values of manipulated variables that maximize the selectivity are determined. If these values correspond to an interior point then the

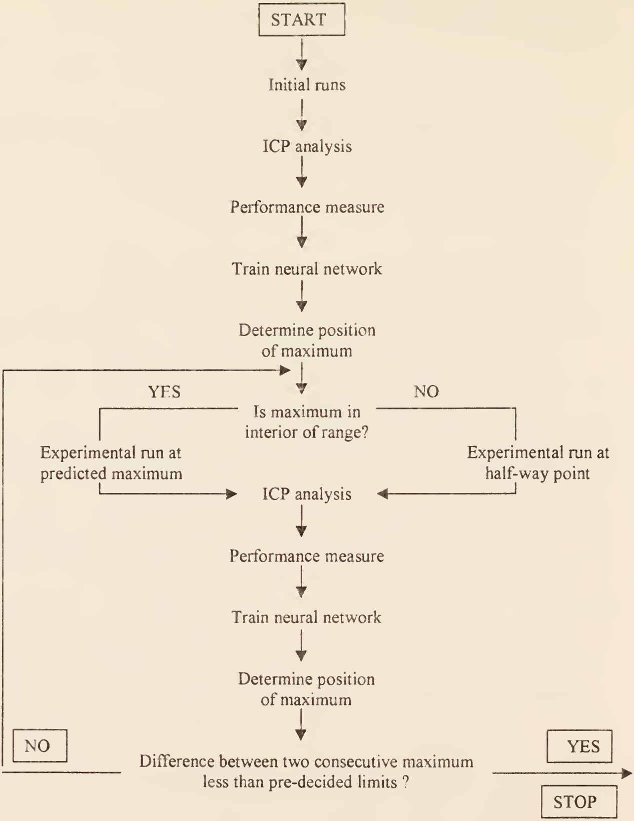
START
Initial runs
ICP analysis
Performance measure
Train neural network
Determine position of maximum
YES
NO
Is maximum in interior of range?
Experimental run at predicted maximum
Experimental run at half-way point
ICP analysis
Performance measure
Train neural network
Determine position of maximum
NO
Difference between two consecutive maximum less than pre-decided limits ?
YES
STOP

Figure 4.2: The run-to-run optimization algorithm

next run will take place at these manipulated variable values; if on the other hand, maximum selectivity is at an exterior point, the next run will be performed at the midpoint between the last run and the predicted optimum values. After completion of the next run, samples are measured for grade and recovery. The new data are subsequently added to the training database and the neural network is retrained. The Optimization algorithm is again used to calculate the new optimal values. These guide the next run, and so on, until convergence is obtained.

The Nelder-Meade method (nonlinear Simplex) can be used to determine the value of the manipulated variables at optimal performance. For three manipulated variables, an initial simplex is defined with four points. This method then takes a series of steps, moving the point of the simplex where the function is lowest through the opposite faces of the simplex to a higher point. These steps are called reflections, and they are constructed to conserve the volume of the simplex. The method expands the simplex in one direction to take larger steps. When it reaches a lower point, it contracts in the traverse direction. This is continued till the decrease in the function value (selectivity) is smaller than some tolerance (1E-3).

4.3 Initial Scattered Experiments

Scattered experiments according to a factorial design were performed to generate data for the initial training of the neural networks. Superficial air velocity, frother concentration, and elutriation water flow rate were selected as the manipulated variables. F-507, which is a non-ionic surfactant, was used as the frother in these experiments.

Experiments were performed with five different levels of superficial air velocity (0.24, 0.42, 0.60, 0.78, and 0.96 cm/s) and frother concentration (5, 10, 15, 20, 15 ppm), and three levels of elutriation water flow rate (9, 10, and 11 gallons per min.).

The design of experiments is shown in Table 4.1. The experiments were designed so as to generate 13 data points, which is the minimum required for training the neural networks, which have three hidden nodes. Experiments were performed according to the design while keeping all other variables constant. After each experiment, three samples from the tailings stream were collected. The samples were then analyzed for %BPL content following the procedure recommended by the Association of Florida Phosphate Chemists (AFPC Analytical Methods, 1980). Since the grade of the feed is known, grade of the concentrate stream can easily be calculated by making a material balance around the column.

4.4 Results and Discussions

The three neural networks of the hybrid model were trained using 13 data points obtained from the designed experiments. The performance of these neural networks is shown in Figures 4.3- 4.5. Figure 4.3 presents the predicted flotation rate constants for phosphate (k_p) against those determined from one-dimensional searches using experimental data as described in chapter two and three. As shown in this figure, the neural network satisfactorily captures the dependence of the flotation rate constant on the selected manipulated variables. Similarly, Figure 4.4 presents the predicted flotation rate constants for gangue (k_g) against those determined from one-dimensional searches using

Table 4.1: Operating conditions for the factorial design

	Frother Concentration	Superficial air velocity	Elutriation water Flow rate
1	-1	-1	-1
2	-1	+1	-1
3	+1	-1	-1
4	+1	+1	+1
5	-1	0	+1
6	+1	0	-1
7	0	-1	+1
8	0	+1	+1
9	0	0	0
10	0	+0.5	0
11	+0.5	0	0
12	0	-0.5	0
13	-0.5	0	0

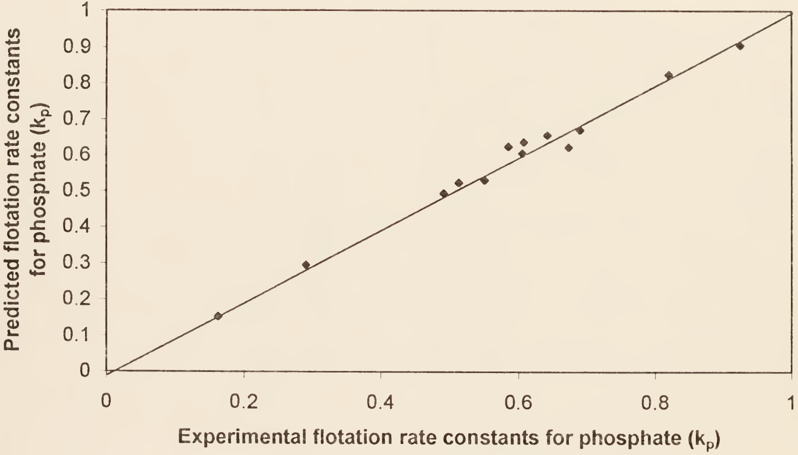

Figure 4.3: Neural network versus experimental flotation rate constant for phosphate (k_p)

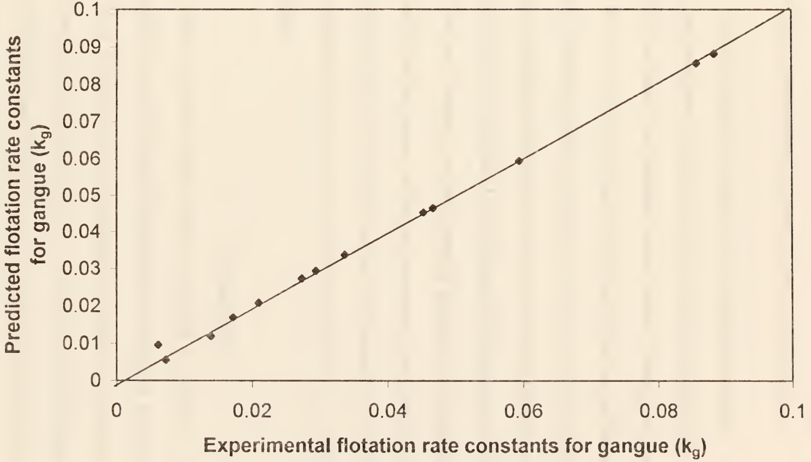

Figure 4.4: Neural network versus experimental flotation rate constant for gangue (k_g)

experimental data. Again, a very good match is seen Figure 4.5 compares the predicted air holdup to the experimental values measured by a differential pressure cell. As shown in this figure, neural network successfully predicts the air holdup.

Table 4.2 shows the results of the 13 designed experiments. As can be seen from this Table, feed flow rate and the %solids content varied significantly. The screw feeder operation was erratic and therefore we were unable to feed at the same rate in each run. Feed flow rate was calculated based on the product flow rate and the tailings flow rate. A specified volume of product and tailings were taken over a period of time (~20 s) and the samples were dried and the weight was taken. In this way, solids flow rate in product and tailings stream were obtained. An overall material balance on the column then gives the feed flow rate. Similarly, an overall material balance on the water phase gives the water flow rate in the feed stream. Solids feed flow rate and the water flow rate in the feed then can be used to obtain the % solids in the feed. Unfortunately, the inability to control feed flow rate and % solids content means that a meaningful run-to-run optimization cannot be conducted.

4.5 Future Work

First, the screw feeder needs to be repaired or replaced. After this has been accomplished, the hybrid model obtained from the designed experiments (Figures 4.3-4.5) should be used with the Nelder-Meade algorithm to determine the experimental conditions of the first optimization run. The results of the run should be analyzed for grade and recovery and these data should be added to the neural network training sets. The networks should then be retrained and the updated hybrid model used to determine

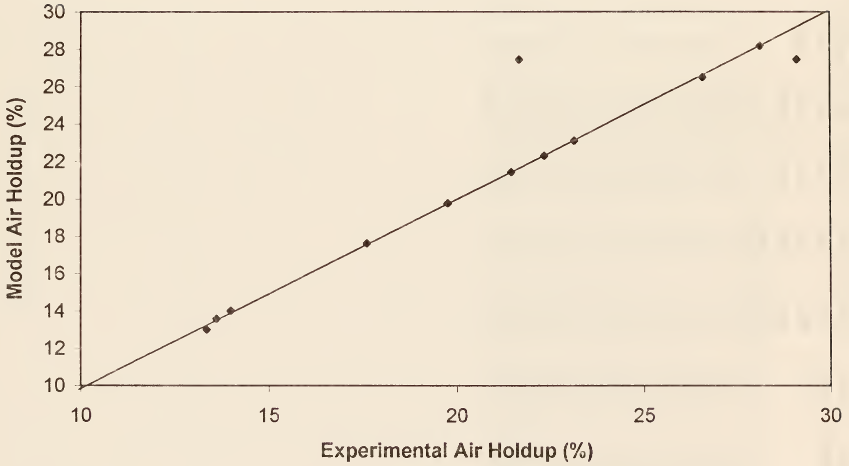

Figure 4.5: Model versus experimental air holdup for frother F-507

Table 4.2: Results of the runs from the factorial design

	Frother conc. (ppm)	Air flow rate (scfm)	Feed flow rate (gpm)	Tailings feed flow rate (gpm)	Elutria-tion flow rate (gpm)	Solids content (%)	Grade (%BPL)	Recovery (%)
1	5	0.0928	0.198	2.014	2.410	59.37	55.95	68.31
2	5	0.3711	0.418	1.779	2.351	35.11	55.36	40.04
3	25	0.0928	0.126	1.432	2.423	48.53	40.07	34.60
4	25	0.3711	0.284	2.062	2.919	35.03	61.86	45.90
5	5	0.2319	0.376	1.897	2.893	49.75	51.04	67.72
6	25	0.2319	0.284	1.650	2 378	47.02	39.59	55.92
7	15	0.0928	0.264	2.355	2.922	36.43	62.68	47.49
8	10	0.3711	0.340	2.275	2.927	41.65	53.99	48.53
9	15	0.2319	0.463	2.173	2.619	38.58	45.63	16.87
10	15	0.3015	0.370	1.838	2.645	42.65	46.26	54.10
11	20	0.2319	0.261	1.694	2.661	49.18	37.50	55.19
12	15	0.1624	0.281	1.853	2.634	42.36	68.40	47.92
13	10	0.2319	0.259	1.758	2.631	41.13	68.03	52.14

the conditions for the next run. This should be repeated with the algorithm of Figure 4.2 until convergence is obtained.

APPENDIX A
CODE FOR THE FIRST PRINCIPLES MODEL FOR ONE LEVEL

```
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include <math.h>
#define GQT 2.5372              //Tailings Flow rate (gallons/min)//
#define GQF 0.6133              //Feed Flow rate (gallons/min)//
#define GQE 3.038               //Elutriation Flow rate (gallons/min)//
#define CS 66.0                 //% Solid in the feed (lb S/lb T)//
#define BPL 24.9                //% BPL of the feed (lb P/lb S) //
#define ROS 2.6                 //Specific Gravity of solids in the feed//
#define DP 122.5                //Particle size in microns //
#define Eg 0.05762              //air hold up //
#define Dia 0.5                 //Diameter of the column (ft.)//
#define L  6.0                  //Height of the column (ft.)//
#define Ku  2.217556            //Flotation rate const. for Phosphate (1/min.)//
#define KGu 0.999965            //Flotation rate const. for Gaunge (1/min.)//
#define Qg  0.1778              //Air flow rate(scfm)//
#define FNF 1.0                 //Feed Location from the top (ft.)//
#define Lf L-FNF

void main1(double,double,double[]);


void main1(double CF, double k, double B[])
{

double QF,QE,QT1,QT,QP,Area,UP,UT,UF,D,DP1,PHIS,USLi,REP,USL,diff;
double a,b,d,alpha,beta,gamma,delta,p,q,m;

QF=0.1336541*GQF;
QE=0.1336541*GQE;
QT1=(0.1336541*GQT);
QP=QF-QT1+QE;
QT=QT1-QE;
Area=0.7853981*Dia*Dia;
UP=QP/Area;
UT=QT/Area;
UF=QF/Area;
```

```
D=12.4*Dia*pow((0.3175*Qg/Area),0.3);

//************** Calculation of slip velocity **************
DP1=DP/1000.0;
PHIS=(CS/100)/((CS/100)+((1-(CS/100))*ROS));
USLi=0.0;
do
{
REP=5.12*DP1*USLi*ROS*(1.0-PHIS);
USL=108.233*DP1*DP1*(ROS-1)*pow((1-PHIS),2.7)/(1+0.15*(pow(REP,0.687)));
diff=USL-USLi;
if(diff<0.0)
diff=-diff;
USLi=USL;
}
while(diff>=0.0001);
USL=(1-Eg)*USL;
//***************** USL calculation ends ***************

a=(UP-USL)/D;
d=(UT+USL)/D;
b=k*(1-Eg)/D;
if(((a*a+4*b)<0.0)||((d*d+4*b)<0.0))
b=0.0;
alpha=(a/2)+(sqrt(a*a+4*b))/2;
beta=(a/2)-(sqrt(a*a+4*b))/2;
gamma=(-d/2)+(sqrt(d*d+4*b))/2;
delta=(-d/2)-(sqrt(d*d+4*b))/2;
//printf("\n alpha=%lf",alpha);
//printf("\n beta=%lf",beta);
//printf("\n gamma=%lf",gamma);
//printf("\n delta=%lf",delta);
//printf("\n delta=%lf",a*a+4*b);

//printf("\n beta*L=%lf",(beta)*(L));
//printf("\n alpha*L=%lf",(alpha)*(L));
//printf("\n gamma*Lf=%lf",(gamma)*(Lf));
//printf("\n delta*Lf=%lf",(delta)*(Lf));
//printf("\n alpha*Lf=%lf",(alpha)*(Lf));
//printf("\n beta*Lf=%lf",(beta)*(Lf));
//p=(((-UP/D)+a-beta)*(exp((beta)*(L))))/(((UP/D)-a+alpha)*(exp((alpha)*(L))));
//q=((-UT/D)+d-delta)/((UT/D)-d+gamma);
if(USL<=UP)
p=((-beta)*exp((beta)*(L)))/(alpha*exp((alpha)*(L)));
```

```
else
p=((-(-beta+a))*exp((beta)*(L)))/((-alpha+a)*exp((alpha)*(L)));
q=-(delta-(QE/(Area*D)))/(gamma-(QE/(Area*D)));
m=(q*exp((gamma)*(Lf))+exp((delta)*(Lf)))/(p*exp((alpha)*(Lf))+exp((beta)*(Lf)));

B[4]=(UF*CF/D)/((m*p*(a-
alpha)*exp((alpha)*(Lf)))+((d+gamma)*q*exp((gamma)*(Lf)))+(m*(a-
beta)*exp((beta)*(Lf)))+((d+delta)*exp((delta)*(Lf))));
//B[3]=(UF*CF/D)/((a+alpha+d+gamma)*exp((gamma)*(Lf)));


B[2]=m*B[4];
B[1]=p*m*B[4];
B[3]=q*B[4];
//B[1]=(exp((gamma-alpha)*Lf))*B[3];
//B[2]=0.0;
//B[4]=0.0;

}

void main(void)
{
double
QF,QE,QT1,QT,QP,Area,DP1,PHIS,USLi,REP,USL,diff,Frank,CF,CFG,K,KG,C,CG;
double RO,ROG,Grade,B[5],BG[5];



QF=0.1336541*GQF;
QE=0.1336541*GQE;
QT1=(0.1336541*GQT);
QP=QF-QT1+QE;
QT=QT1-QE;
Area=0.7853981*Dia*Dia;
printf("\n UP=%lf",QP/Area);

//************** Calculation of slip velocity **************
DP1=DP/1000.0;
PHIS=(CS/100)/((CS/100)+((1-(CS/100))*ROS));
USLi=0.0;
do
{
REP=5.12*DP1*USLi*ROS*(1.0-PHIS);
USL=108.233*DP1*DP1*(ROS-1)*pow((1-PHIS),2.7)/(1+0.15*(pow(REP,0.687)));
diff=USL-USLi;
if(diff<0.0)
diff=-diff;
USLi=USL;
}
while(diff>=0.0001);
USL=(1-Eg)*USL;
printf("\n USL=%lf",USL);
//***************** USL calculation ends ***************

Frank=BPL/0.733;
CF=(Frank/100.0)*(CS/100.0)*ROS*62.41818/((CS/100.0)+(1.0-(CS/100.0))*ROS);
CFG=(1.0-(Frank/100.0))*(CS/100.0)*ROS*62.41818/((CS/100.0)+(1.0-
(CS/100.0))*ROS);

K=Ku;KG=KGu;

main1(CF,K,B);

C=B[3]+B[4];

RO=(((QF*CF)-((QT1+Area*USL)*C))/(QF*CF))*100.0;

main1(CFG,KG,BG);

CG=BG[3]+BG[4];

ROG=(((QF*CFG)-((QT1+Area*USL)*CG))/(QF*CFG))*100.0;
Grade=((QF*CF-(QT1+Area*USL)*C)/(((QF*CF-(QT1+Area*USL)*C)+(QF*CFG-
(QT1+Area*USL)*CG)))*100.0;
Grade=Grade*0.733;
printf("\n C=%lf",C);
printf("\n CG=%lf",CG);
printf("\n CF=%lf",CF);
printf("\n CFG=%lf",CFG);

printf("\n Overall Recovery=%.1lf %",RO);
printf("\n ROG=%.1lf %",ROG);
printf("\n GRADE=%.1lf %",Grade);
}


#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include <math.h>
#define GQT 2.2551          //Tailings Flow rate (gallons/min)//
#define GQF 0.9318          //Feed Flow rate (gallons/min)//
```

```
#define GQE 2.6948          //Elutriation Flow rate (gallons/min)//
#define CS 66.0             //% Solid in the feed (lb S/lb T)//
#define BPL 24.9            //% BPL of the feed (lb P/lb S) //
#define ROS 2.6             //Specific Gravity of solids in the feed//
#define DP 122.5            //Particle size in microns //
#define Eg 0.115            //air hold up //
#define Dia 0.5             //Diameter of the column (ft.)//
#define L  6.0              //Height of the column (ft.)//
#define KGu 0.0             //Flotation rate const. for Gaunge (1/min.)//
#define Qg  0.1778          //Air flow rate(scfm)//
#define FNF 1.0             //Feed Location from the top (ft.)//
#define Lf L-FNF
#define ES 1E-3

void main1(double,double,double[]);
double model(double);


void main(void)
{

double kl,yl,gl,ku,yu,gu,kr[3],yr,gr,R[3],a,EA,Test,x;
double Grade,Grade_feed,Grade_prod;
int i;

R[1]=70.915;
Grade=25.8;
Grade_feed=BPL/73.3;
Grade_prod=Grade/73.3;
R[2]=R[1]*Grade_feed*(1-Grade_prod)/((1-Grade_feed)*Grade_prod);
printf("\nComponent[1]=phosphate");
printf("\nComponent[2]=gangue");

for(i=1;i<=2;i++)
{
kl=0.0;
yl=model(kl);
gl=R[i]-yl;
if(gl>0.0)
{

do
{
printf("\nEnter an initial guess for flotation rate constant for component[%d]",i);
printf("\nku=");
scanf("\n %lf", &ku);
yu=model(ku);
gu=R[i]-yu;
a=gl*gu;
}
while(a>=0);


EA=1.1*ES;
while(EA>ES)
        {
        kr[i]=ku-(gu*(kl-ku))/(gl-gu);
        x=kr[i];
        if((kl+ku)!=0)
                EA=fabs((ku-kl)/(kl+ku))*100;
        yr=model(x);
        gr=R[i]-yr;
        Test=gl*gr;
        if(Test==0.0)
                EA=0;
        else if(Test<0.0)
                ku=kr[i];
        else if(Test>0.0)
                kl=kr[i];
        printf("\nkr[%d]=%lf",i,kr[i]);
        }

printf("\nFlotation rate constant for component[%d]=%lf",i,kr[i]);
getch();
}
else
printf("\nkr[%d]=0.0",i);
}

}


void main1(double CF, double k, double B[])
{

double QF,QE,QT1,QT,QP,Area,UP,UT,UF,D,DP1,PHIS,USLi,REP,USL,diff;
double a,b,d,alpha,beta,gamma,delta,p,q,m;

QF=0.1336541*GQF;
QE=0.1336541*GQE;
QT1=(0.1336541*GQT);
```

```
QP=QF-QT1+QE;
QT=QT1-QE;
Area=0.7853981*Dia*Dia;
UP=QP/Area;
UT=QT/Area;
UF=QF/Area;


D=12.4*Dia*pow((0.3175*Qg/Area),0.3);

//************** Calculation of slip velocity **************
DP1=DP/1000.0;
PHIS=(CS/100)/((CS/100)+((1-(CS/100))*ROS));
USLi=0.0;
do
{
REP=5.12*DP1*USLi*ROS*(1.0-PHIS);
USL=108.233*DP1*DP1*(ROS-1)*pow((1-PHIS),2.7)/(1+0.15*(pow(REP,0.687)));
diff=USL-USLi;
if(diff<0.0)
diff=-diff;
USLi=USL;
}
while(diff>=0.0001);
USL=(1-Eg)*USL;
//***************** USL calculation ends ***************

a=(UP-USL)/D;
d=(UT+USL)/D;
b=k*(1-Eg)/D;
if(((a*a+4*b)<0.0)||((d*d+4*b)<0.0))
b=0.0;
alpha=(a/2)+(sqrt(a*a+4*b))/2;
beta=(a/2)-(sqrt(a*a+4*b))/2;

gamma=(-d/2)+(sqrt(d*d+4*b))/2;
delta=(-d/2)-(sqrt(d*d+4*b))/2;
//printf("\n alpha=%lf",alpha);
//printf("\n beta=%lf",beta);
//printf("\n gamma=%lf",gamma);
//printf("\n delta=%lf",delta);

//printf("\n beta*L=%lf",(beta)*(L));
//printf("\n alpha*L=%lf",(alpha)*(L));
//printf("\n gamma*Lf=%lf",(gamma)*(Lf));
//printf("\n delta*Lf=%lf",(delta)*(Lf));
```

```
//printf("\n alpha*Lf=%lf",(alpha)*(Lf));
//printf("\n beta*Lf=%lf",(beta)*(Lf));
//p=(((-UP/D)+a-beta)*(exp((beta)*(L))))/(((UP/D)-a+alpha)*(exp((alpha)*(L))));
//q=((-UT/D)+d-delta)/((UT/D)-d+gamma);
if(USL<=UP)
p=((-beta)*exp((beta)*(L)))/(alpha*exp((alpha)*(L)));
else
p=((-(-beta+a))*exp((beta)*(L)))/((-alpha+a)*exp((alpha)*(L)));
q=-(delta-(QE/(Area*D)))/(gamma-(QE/(Area*D)));
m=(q*exp((gamma)*(Lf))+exp((delta)*(Lf)))/(p*exp((alpha)*(Lf))+exp((beta)*(Lf)));

B[4]=(UF*CF/D)/((m*p*(a-
alpha)*exp((alpha)*(Lf)))+((d+gamma)*q*exp((gamma)*(Lf)))+(m*(a-
beta)*exp((beta)*(Lf)))+((d+delta)*exp((delta)*(Lf))));
//B[3]=(UF*CF/D)/((a+alpha+d+gamma)*exp((gamma)*(Lf)));


B[2]=m*B[4];
B[1]=p*m*B[4];
B[3]=q*B[4];
//B[1]=(exp((gamma-alpha)*Lf))*B[3];
//B[2]=0.0;
//B[4]=0.0;

}

double model(double Ku)
{
double
QF,QE,QT1,QT,QP,Area,DP1,PHIS,USLi,REP,USL,diff,Frank,CF,CFG,K,KG,C,CG;
double RO,ROG,Grade,B[5],BG[5];



QF=0.1336541*GQF;
QE=0.1336541*GQE;
QT1=(0.1336541*GQT);
QP=QF-QT1+QE;
QT=QT1-QE;
Area=0.7853981*Dia*Dia;
//printf("\n UP=%lf",QP/Area);

//************** Calculation of slip velocity **************
DP1=DP/1000.0;
PHIS=(CS/100)/((CS/100)+((1-(CS/100))*ROS));
USLi=0.0;
```

```
do
{
REP=5.12*DP1*USLi*ROS*(1.0-PHIS);
USL=108.233*DP1*DP1*(ROS-1)*pow((1-PHIS),2.7)/(1+0.15*(pow(REP,0.687)));
diff=USL-USLi;
if(diff<0.0)
diff=-diff;
USLi=USL;
}
while(diff>=0.0001);
USL=(1-Eg)*USL;
//printf("\n USL=%lf",USL);
//***************** USL calculation ends ***************

Frank=BPL/0.733;
CF=(Frank/100.0)*(CS/100.0)*ROS*62.41818/((CS/100.0)+(1.0-(CS/100.0))*ROS);
CFG=(1.0-(Frank/100.0))*(CS/100.0)*ROS*62.41818/((CS/100.0)+(1.0-
(CS/100.0))*ROS);

K=Ku;KG=KGu;

main1(CF,K,B);

C=B[3]+B[4];

RO=(((QF*CF)-((QT1+Area*USL)*C))/(QF*CF))*100.0;

main1(CFG,KG,BG);

CG=BG[3]+BG[4];

ROG=(((QF*CFG)-((QT1+Area*USL)*CG))/(QF*CFG))*100.0;
Grade=((QF*CF-(QT1+Area*USL)*C)/((QF*CF-(QT1+Area*USL)*C)+(QF*CFG-
(QT1+Area*USL)*CG)))*100.0;
Grade=Grade*0.733;
//printf("\n C=%lf",C);
//printf("\n CG=%lf",CG);
//printf("\n CF=%lf",CF);
//printf("\n CFG=%lf",CFG);

//printf("\n Overall Recovery=%.1lf %",RO);
//printf("\n ROG=%.1lf %",ROG);
//printf("\n GRADE=%.1lf %",Grade);
return (RO);
}
```

APPENDIX B
CODE FOR THE FIRST PRINCIPLES MODEL FOR TWO LEVELS

```
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include <math.h>
#define GQT 1.5249          //Tailings Flow rate (gallons/min)//
#define GQF 0.8858          //Feed Flow rate (gallons/min)//
#define GQE 2.5102          //Elutriation Flow rate (gallons/min)//
#define CS 67 2             //% Solid in the feed (lb S/lb T)//
#define BPL 18.2            //% BPL of the feed (lb P/lb S) //
#define ROS 2.6             //Specific Gravity of solids in the feed//
#define DP 208.25           //Particle size in microns //
#define Eg 0.1487           //air hold up //
#define Dia 0.5             //Diameter of the column (ft.)//
#define L  6.0              //Height of the column (ft.)//
#define Ku  6.042797        //Flotation rate const. for Phosphate (1/min.)//
#define KGu 0.048094        //Flotation rate const. for Gaunge (1/min.)//
#define Qg  0.2706          //Air flow rate(scfm)//
#define N 15
#define FNF 1.0             //Feed Location from the top (ft.)//
#define DELT 0.1
#define n N+1
#define a1  1.0             //a1=0-->explicit;a1=1-->implicit//


void main1(double,double,double,double[]);


void main1(double CF,double K,double D,double C[])
{

static double A[n][n],V[n],S[n];
double A1,A2,A3,A4,A5,A6,A7,A8,A9,UP,UT,UT1,UF,DELZ,QP,QF,QT,QE,QT1;
double A10,A11,A12,A13,A14;
int O[n];
int i,j,k,ii,Pivot,IDummy,NF;
double Big, Dummy,factor,Sum,Area,USLi,USL,REP,diff,PHIS,DP1;
        QF=0.1336541*GQF;
        QE=0.1336541*GQE;
        QT1=(0.1336541*GQT);
        QP=QF-QT1+QE;
        QT=QT1-QE;
        DELZ=(L/N);
        NF=((FNF/DELZ)+1)*1;
        Area=0.7853981*Dia*Dia;


//************** Calculation of slip velocity **************
DP1=DP/1000.0;
PHIS=(CS/100)/((CS/100)+((1-(CS/100))*ROS));
USLi=0.0;
do
{
REP=5.12*DP1*USLi*ROS*(1.0-PHIS);
USL=108.233*DP1*DP1*(ROS-1)*pow((1-PHIS),2.7)/(1+0.15*(pow(REP,0.687)));
diff=USL-USLi;
if(diff<0.0)
diff=-diff;
USLi=USL;
}
while(diff>=0.0001);
USL=(1-Eg)*USL;
//***************** USL calculation ends ***************
//USL=0.0;

if(USL<=(QP/Area))
{
        UP=QP/Area;UP=(UP-USL)/(1-Eg);
        UT=QT/Area;UT=(UT+USL)/(1-Eg);UT1=QT1/Area;UT1=(UT1+USL)/(1-Eg);
        UF=QF/Area;UF=UF/(1-Eg);
        A3=D/(DELZ*DELZ);
        A4=(-UP-(2.0*D/DELZ)-(K*DELZ))/(DELZ);
        A5=(UP+(D/DELZ))/DELZ;
        A1=A3+A4;
        A2=A5;
        A6=(UF*CF)/DELZ;
        A7=A3;
        A10=(UT/DELZ)+A3;
        A8=A3+A4-A10;
        A9=A3;
        A11=(-UT-(2.0*D/DELZ)-(K*DELZ))/(DELZ);
        A12=A3;
        A13=A10;
        //A14=A3+A11;
  A14=(-UT1-(D/DELZ)-(K*DELZ))/(DELZ);
}
else
{
        UP=QP/Area;UP=(UP-USL)/(1-Eg);UP=-UP;
        UT=QT/Area;UT=(UT+USL)/(1-Eg);UT1=QT1/Area;UT1=(UT1+USL)/(1-Eg);
        UF=QF/Area;UF=UF/(1-Eg);
        A3=(UP+(D/DELZ))/DELZ;
        A4=(-UP-(2.0*D/DELZ)-(K*DELZ))/(DELZ);
        A5=D/(DELZ*DELZ);
        A1=A4+A5;
        A2=A5;
        A6=(UF*CF)/DELZ;
        A7=A3;
        A11=(-UT-(2.0*D/DELZ)-(K*DELZ))/(DELZ);
        A8=A11;
        A9=A5;
        A10=(UT/DELZ)+A5;
        A12=A5;
        A13=A10;
        //A14=A5+A11;
  A14=(-UT1-(D/DELZ)-(K*DELZ))/(DELZ);
}

//*********Definition of Row=1***************************
        A[1][1]=1.0-(a1*DELT*A1);
        A[1][2]=-(a1*DELT*A2);
        for(i=3;i<=N;i++)
                A[1][i]=0.0;
//***********Definition of Row=2 to NF-1 *****************
        for(i=2;i<NF;i++)
        {
                A[i][i-1]=-(a1*DELT*A3);
                A[i][i]=1.0-(a1*DELT*A4);
                A[i][i+1]=-(a1*DELT*A5);
                for(j=1;j<i-1;j++)
                        A[i][j]=0.0;
                for(j=i+2;j<=N;j++)
                        A[i][j]=0.0;
        }
//***********Definition of Row=NF *************************
        A[NF][NF-1]=-(a1*DELT*A7);
        A[NF][NF]=1.0-(a1*DELT*A8);
        A[NF][NF+1]=-(a1*DELT*A9);
        for(i=1;i<NF-1;i++)
                        A[NF][i]=0.0;
        for(i=NF+2;i<=N;i++)
```

```
            A[NF][i]=0.0;
//**************Definition of Row=NF+1 to N-1 *************
    for(i=NF+1;i<N;i++)
    {
        A[i][i-1]=-(a1*DELT*A10);
        A[i][i]=1.0-(a1*DELT*A11);
        A[i][i+1]=-(a1*DELT*A12);
        for(j=1;j<i-1;j++)
            A[i][j]=0.0;
        for(j=i+2;j<=N;j++)
            A[i][j]=0.0;
    }
//*************Definition of Row=N ***********************
    A[N][N-1]=-(a1*DELT*A13);
    A[N][N]=1.0-(a1*DELT*A14);
    for(i=1;i<N-1;i++)
        A[N][i]=0.0;
//*********Row Definition ends *****************************

//************* Defifnition of column vector *******************
V[1]=(1.0+(1.0-a1)*DELT*A1)*C[1]+(1.0-a1)*DELT*A2*C[2];
//printf("\n V[1]=%lf \n",V[1]);
//getch();
for(i=2;i<NF;i++)
    V[i]=(1.0+(1.0-a1)*DELT*A4)*C[i]+(1.0-a1)*DELT*A3*C[i-1]+(1.0-
a1)*DELT*A5*C[i+1];
    //printf("\n V[%d]=%lf \n",i,V[i]);
    //getch();}
V[NF]=(DELT*A6)+(1.0+(1.0-a1)*DELT*A8)*C[NF]+(1.0-a1)*DELT*A7*C[NF-
1]+(1.0-a1)*DELT*A9*C[NF+1];
//printf("\n V[NF]=%lf \n",V[NF]);
//getch();
for(i=NF+1;i<N;i++)
    V[i]=(1.0+(1.0-a1)*DELT*A11)*C[i]+(1.0-a1)*DELT*A10*C[i-1]+(1.0-
a1)*DELT*A12*C[i+1];
    //printf("\n V[%d]=%lf \n",i,V[i]);
    //getch();}
V[N]=(1.0+(1.0-a1)*DELT*A14)*C[N]+(1.0-a1)*DELT*A13*C[N-1];
//printf("\n V[N]=%lf \n",V[N]);
//getch();

//**************Definition of column vector ends **********


//*************** ORDERING ******************************
```

```
for(i=1;i<=N;i++)
{
        O[i]=i;
        S[i]=abs(A[i][1]);
                for(j=2;j<=N;j++)
                {
                if(abs(A[i][j])>S[i])
                        S[i]=abs(A[i][j]);
                }
}
//*************** Ordering ends *******************************

//***************Gauss Elimination *************************

for(k=1;k<N;k++)
{
                                //****  Pivoting   *********//

Pivot=k;
Big=abs(A[O[k]][k]/S[O[k]]);
        for(ii=k+1;ii<=N;ii++)
        {
        Dummy=abs(A[O[ii]][k]/S[O[ii]]);
                if(Dummy>Big)
                {       Big=Dummy;
                        Pivot=ii;
                }
        }
IDummy=O[Pivot];
O[Pivot]=O[k];
O[k]=IDummy;
                                //*** End Pivoting*******//


for(i=k+1;i<=N;i++)
{
        factor=A[O[i]][k]/A[O[k]][k];
                for(j=k+1;j<=N;j++)
                {
                A[O[i]][j]=A[O[i]][j]-(factor*A[O[k]][j]);
                }
        V[O[i]]=V[O[i]]-(factor*V[O[k]]);
}
}
//***************Gauss Elimination ends ******************
```

```
//************* Substitution ******************************

C[N]=V[O[N]]/A[O[N]][N];
        for(i=N-1;i>=1;i--)
        {
        Sum=0.0;
                for(j=i+1;j<=N;j++)
                {
                Sum=Sum+(A[O[i]][j]*C[j]);
                }
        C[i]=(V[O[i]]-Sum)/A[O[i]][i];
        }
}

//*****************  Substitution ends  ********************
//************************************************************
//$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
//&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&
&&&&&&&&&&&&
//############################################################
//!!!!!!!!!!!!!!!!   Main1 Ends   !!!!!!!!!!!!!!!!!!!!!!!!

void main(void)
{
double
Grade,CF,CFG,K,KG,D,RO,Area,Z,RT,QP,QT,QF,TIME,Pe,Tp,a,b1,b2,REQNP,REQN
G;
int l,i,FLAG;//NF;
double C[n],CG[n],USLi,USL,REP,diff,PHIS,Gradeqn,c,d1,d2,Frank;//DELZ;
double Du,QE,QT1,R,DP1,Selectivity,Selecteqn,ROG,RG;//RC;
QF=0.1336541*GQF;
QE=0.1336541*GQE;
QT1=(0.1336541*GQT);
QP=QF-QT1+QE;
QT=QT1-QE;
//DELZ=(L/N);
//NF=((FNF/DELZ)+1)*1;
Area=0.7853981*Dia*Dia;
printf("\n UP=%lf",QP/Area);
//************** Calculation of slip velocity **************
DP1=DP/1000.0;
PHIS=(CS/100)/((CS/100)+((1-(CS/100))*ROS));
USLi=0.0;
do
{
REP=5.12*DP1*USLi*ROS*(1.0-PHIS);
```

```
USL=108.233*DP1*DP1*(ROS-1)*pow((1-PHIS),2.7)/(1+0.15*(pow(REP,0.687)));
diff=USL-USLi;
if(diff<0.0)
diff=-diff;
USLi=USL;
}
while(diff>=0.0001);
USL=(1-Eg)*USL;
printf("\n USL=%lf",USL);
//****************** USL calculation ends ****************
//USL=0.0;
//printf("\nEnter your initial values for Phosphate concentration:\n");
for(i=1;i<=N;i++)
{
C[i]=0.0;
//printf("\n C[%d]=",i);
//scanf("%lf",&C[i]);
}
//printf("\nEnter your initial values for Gaunge concentration:\n");
for(i=1;i<=N;i++)
{
CG[i]=0.0;
//printf("\n CG[%d]=",i);
//scanf("%lf",&CG[i]);
}
Frank=BPL/0.733;
CF=(Frank/100.0)*(CS/100.0)*ROS*62.41818/((CS/100.0)+(1.0-(CS/100.0))*ROS);
CFG=(1.0-(Frank/100.0))*(CS/100.0)*ROS*62.41818/((CS/100.0)+(1.0-
(CS/100.0))*ROS);
printf("\n CF=%lf",CF);
printf("\n CFG=%lf",CFG);
//getch();
Du=12.4*Dia*pow((0.3175*Qg/Area),0.3); // ft2/min//
K=Ku;KG=KGu;D=Du/(1-Eg);
RT=L*Area*(1-Eg)/(QF+(L*Area*(1-Eg)*abs(KG)));
Z=(50.0*RT/DELT)-1;
TIME=DELT;


///*****$$$$$%%%%%@@@@@@ HERE IS THE FLAG BETWEEN STEADY-
STATE & DYNAMIC
FLAG=1;  //1 for dynamic, other values for steady-state approximation
if(FLAG==1)
{
for(l=1;l<=Z;l++)
{
```

```
main1(CF,K,D,C);
//printf("\n Following are the values of Phosphate C[i] at Time=%lf RT. \n",TIME/RT);
        //for(i=1;i<=N;i++)
        //printf("\n C[%d]=%lf",i,C[i]);
        //getch();
RO=(((QF*CF)-((QT1+Area*USL)*(C[N])))/(QF*CF))*100.0;

//RC=(((QF*CF)-(QP*(C[NF]))-(QT1*(C[N])))/((QF*CF)-(QP*(C[NF]))))*100.0;
//RC needs modification in terms of QP and QT1 (i.e. has to include USL)
//printf("\n The overall recovery at Time=%lf RT is RO=%lf%.",TIME/RT,RO);
//printf("\n The collection zone recovery at Time=%lf RT is
RC=%lf%.\n",TIME/RT,RC);
//getch();

main1(CFG,KG,D,CG);
//printf("\n Following are the values of Gaunge CG[i] at Time=%lf RT. \n",TIME/RT);
        //for(i=1;i<=N;i++)
        //printf("\n CG[%d]=%lf",i,CG[i]);
        //getch();
ROG=(((QF*CFG)-((QT1+Area*USL)*(CG[N])))/(QF*CFG))*100.0;
Grade=((QF*CF-(QT1+Area*USL)*C[N])/((QF*CF-
(QT1+Area*USL)*C[N])+(QF*CFG-(QT1+Area*USL)*CG[N])))*100.0;
Grade=Grade*0.733;
//printf("\n The grade at Time=%lf RT is %lf",TIME/RT,Grade);
//getch();

Selectivity=RO-ROG;

TIME=(l+1)*DELT;
}
//printf("\n G100=%lf",G100);
//for(i=1;i<=N;i++)
        //printf("\n CG[%d]=%lf",i,CG[i]);
        //getch();
printf("\n CG[N]=%lf",CG[N]);
//printf("\n C[N]=%lf",C[N]);
//printf("\n C[1]=%lf",C[1]);
printf("\n C[N]=%lf",C[N]);
printf("\n Overall Recovery=%.1lf %",RO);
printf("\n GRADE=%.1lf %",Grade);
printf("\n Sep_eff=%.1lf ",Selectivity);
printf("\n ROG=%.1lf %",ROG);
//printf("\n R=%.1lf %",(1-(C[N]/CF))*100);
getch();
}
else
```

```
{
//******************** Steady state Recovery calculation******************
Pe=(((QT/Area)+USL)*(L))/(D*(1-Eg));
//printf("\n Pe=%lf",Pe);
Tp=((L)*(1-Eg))/((QT/Area)+USL);
//printf("\n Tp=%lf",Tp);
a=sqrt(1+(4*K*Tp/Pe));
//printf("\n a=%lf",a);
//printf("\n K*Tp=%lf",K*Tp);
b1=exp(a*Pe/2);
b2=exp(-a*Pe/2);
REQNP=(1-((4*a*exp(Pe/2))/(((1+a)*(1+a)*b1)-((1-a)*(1-a)*b2))))*100;
R=(1-((1-(REQNP/100))*((QT+Area*USL)/QF)))*100;
c=sqrt(1+(4*KG*Tp/Pe));
d1=exp(c*Pe/2);
d2=exp(-c*Pe/2);
REQNG=(1-((4*c*exp(Pe/2))/(((1+c)*(1+c)*d1)-((1-c)*(1-c)*d2))))*100

;
RG=(1-((1-(REQNG/100))*((QT+Area*USL)/QF)))*100;
//printf("\n REQNP=%.1lf%",REQNP);
printf("\n Recovery=%.1lf%",R);
//printf("\n REQNG=%lf%",REQNG);
Gradeqn=((QF*CF-(QT+Area*USL)*(1-(REQNP/100))*CF)/((QF*CF-
(QT+Area*USL)*(1-(REQNP/100))*CF)+(QF*CFG-(QT+Area*USL)*(1-
(REQNG/100))*CFG)))*100.0;

Selecteqn=R-RG;

printf("\n GRADEQN=%.1lf %",Gradeqn);
printf("\n Sep_eff=%.1lf ",Selecteqn);
getch();
}
//*********Steady state Recovery calculation ends ***********
}
```

LIST OF REFERENCES

Association of Florida Phosphate Chemists, *AFPC Analytical Methods*, 6th ed. (1980).

Bhat, N. and T.J. McAvoy, "Use of Neural Nets for Dynamic Modeling and Control of Chemical Process Systems," *Comput. Chem. Eng.*, 14:573 (1990).

Boutin, P. and D.A. Wheeler, "Column Flotation," Mining World, 20(3): 47-50 (1967).

Chapra, S.C. and R. P. Canale, *Numerical Methods for Engineers*, McGraw Hill, New York (1988).

Cubillos, F., P. Alvarez, J. Pinto, and E. Lima, "Hybrid-Neural Modeling for Particulate Solid Drying Processes," *Powder Technology*, 87, p.153 (1996).

Cubillos, F.A., and E. L. Lima, "Identification and Optimizing Control of a Rougher Flotation Circuit Using an Adaptable Hybrid-Neural Model," *Minerals Engineering*, 10(7): 707-721 (1997).

Dobby, G.S., and J. A. Finch, "Column Flotation: A Selected Review, Part II," *Mineral Engineering*, 4: 911-923 (1991).

Dobby, G.S., and J. A. Finch, "Mixing Characteristics of Industrial Flotation Columns," *Chem. Eng. Sci.*, 40: 1061-1068 (1985).

Finch, J. A., and G. S. Dobby, *Column Flotation*, Pergamon Press, Toronto (1990).

Hertz, J., A. Krogh, and R.G. Palmer, *Introduction to the Theory of Neural Computations*, Addison-Wesley Publishing Company, Redwood City, CA, 5th ed. (1992).

Himmelblau, D.M., *Applied Nonlinear Programming*, McGraw Hill, New York (1972).

Hornik, K.M., M. Stinchcombe, and H. White, "Multi-layer Feedforward Networks Are Universal Approximators," *Neural Networks*, 2:359 (1989).

Johansen, T. A., and B. A. Foss, "Representing and Learning Unmodeled Dynamics with Neural Network Memories," *Proc. Am. Control Conf.*, Chicago, 3:3037-3037 (1992).

Kirkpatrick, S., Jr. C. D. Gelatt, and M. P. Vecchi, "Optimization by Simulated Annealing," *Science*, 220:671-680 (1983).

Kramer, M.A., and M. L. Thompson, "Embedding Theoretical Models in Neural Networks," *Proc. Am. Control Conf.*, Chicago, 1:475-479 (1992).

Liu, P.-H., T. Potter, S. A. Svoronos, and B. Koopman, "Hybrid Model of Nitrogen Dynamics in a Periodic Wastewater Treatment Process," *AIChE Annual Meeting*, Paper No. 195an (1995).

Luttrell, G.H., and R. H. Yoon, "A Flotation Column Simulator Based on Hydrodynamic Principle," *Inter. J. Miner. Process.*, 33:355-368 (1992).

Luttrell, G.H., and R. H. Yoon, "Column Flotation-A Review," in *Beneficiation of Phosphate: Theory and Practice*, Eds. H. El-Shall, B.M. Moudgil and R. Wiegel, SME, Littleton, Colorado, 361-369 (1993).

Masters, T., *Practical Neural Network Recipes in C++*, Academic Press, New York (1993).

Mavros, P., "Mixing in Flotation Columns. Part 1: Axial Dispersion Modeling," *Mineral Engineering*, 6: 465-478 (1993).

Perry, R.H., D. W. Green, and J. O. Maloney, *Perry's Chemical Engineers Handbook*, McGraw-Hill Book Company, New York, 6th ed. (1984).

Polak, E., *Computational Methods in Optimization*, Academic Press, New York (1971).

Psichogios, D.C., and L. H. Ungar, "Process Modeling using Structured Neural Networks," *Proc. Am. Control Conf.*, Chicago, 3:1917-1921 (1992a).

Psichogios, D.C., and L. H. Ungar, "A Hybrid Neural-Network First Principles Approach to Process Modeling," *AIChE J.*, 38:1499-1511 (1992b).

Reuter, M., J. Van Deventer, and P. Van Der Walt, "A Generalized Neural-Net Rate Equation," *Chem. Eng. Sci.*, 48:1281 (1993).

Rumelhart, D., and J. McClelland, *Parallel Distributed Processing*, MIT Press, Cambridge, MA (1986).

Sastry, K.V.S., and K. D. Loftus, "Mathematical Molding and Computer Simulation of Column Flotation," in *Column Flotation'88*, Ed. K.V.S. Sastry, SME-AIME, Littleton, Colorado, 57-68 (1988).

Su, H.-T., P. A. Bhat, P. A. Minderman, and T. J. McAvoy, "Integrating Neural Networks with First Principles Models for Dynamic Modeling," *IFAC Symp. on*

Dynamics and Control of Chemical Reactors, Distillation Columns, and Batch Processes, DYCORD+, (1992).

Thompson, M.L., and M. A. Kramer, "Modeling Chemical Processes Using Prior Knowledge and Neural Networks," *AIChE J.*, 40(8):1328-1340 (1994).

Villeneuve, J., M.-V Durance, C. Guillaneau, A.N. Santana, R.V.G. da Silva, and M.A.S. Martin, "Advanced Use of Column Flotation Models for Process Optimization," in *COLUMN'96*, Eds. Gomez, C.O. and J.A. Finch, The Metallurgical Society of the Canadian Institute of Mining, Metallurgy and Petroleum, 51-62 (1996).

Wheeler D.A., "Historical View of Column Flotation Development," in *Column Flotation'88*, Ed. K.V.S. Sastry, SME-AIME, Littleton, Colorado, 3-4 (1988).

Xu, M., and J.A. Finch, "The Axial Dispersion Model in Flotation Column Studies," *Mineral Engineering*, 4: 553-562 (1991).

Yeager, D., C.L. Karr, and D.A. Stanley, "Column Flotation Model Tuning Using a Genetic Algorithm," *SME Annual Meeting*, Preprint 95-204 (1995).

Yianatos, J.B., J.A. Finch, G.S. Dobby, and M. Xu, "Bubble Size Estimation in a Bubble Swarm," *J. Colloid Interface Sci.*, 126(1):37-44 (1988).

Yoon, R.H., G.H. Luttrell, and G.T. Adel, "Advances in Fine Particle Flotation," in *Challenges in Mineral Processing*, Eds. K.V.S. Sastry and M.C. Fuerstenau, SME-AIME, Littleton, Colorado, 487-506 (1989).

Yoon, R.H., G.H. Luttrell, G.T. Adel, and M.J. Mankosa, "Recent Advances in Fine Coal Flotation," in *Advances in Coal and Mineral Processing Using Flotation*, Eds. S. Chandler and R.R. Klimpel, SME-AIME, Littleton, Colorado, 211-218 (1988).

BIOGRAPHICAL SKETCH

Sanjay Gupta obtained a Bachelor of Engineering degree in chemical engineering from the University of Roorkee, Roorkee, India in May 1993. He then worked as a process engineer at Burmah-Shell Refinery Ltd., Bombay, India, for one year. He continued his higher studies at Drexel University, Philadelphia, and obtained a Master of Science degree in chemical engineering in March 1997. He joined the Department of Chemical engineering at University of Florida to pursue his Ph.D. degree in August, 1996.

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Spyros A. Svoronos

Spyros A. Svoronos, Chairman
Professor of Chemical Engineering

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Hassan El-Shall, Cochairman
Engineer of Materials Science and
Engineering

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Richard Dickinson
Assistant Professor of Chemical Engineering

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Ben Koopman
Professor of Environmental Engineering
Sciences

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Oscar Crisalle

Oscar Crisalle
Associate Professor of Chemical Engineering

This dissertation was submitted to the Graduate Faculty of the College of Engineering and to the Graduate School and was accepted as partial fulfillment of the requirements for the degree of Doctor of Philosophy.

May 1999

Winfred M Phillips

Winfred M. Phillips
Dean, College of Engineering

M. J. Ohanian
Dean, Graduate School